ClearPath OS 2200 TeamQuest Baseline® and TeamQuest Online®

USER GUIDE

Copyright © 2013 TeamQuest Corporation. All Rights Reserved.

February 2013 Release Level 7R4G TQ–01011.18

The names, places and/or events used in this publication are purely fictitious and are not intended to correspond to any real individual, group, company or event. Any similarity or likeness to any real individual, company or event is purely coincidental and unintentional.

NO WARRANTIES OF ANY NATURE ARE EXTENDED BY THE DOCUMENT. Any product and related material disclosed herein are only furnished pursuant and subject to the terms and conditions of a license agreement. The only warranties made, remedies given, and liability accepted by TeamQuest, if any, with respect to the products described in this document are set forth in such license agreement. TeamQuest cannot accept any financial or other responsibility that may be the result of your use of the information in this document or software material, including direct, indirect, special, or consequential damages.

You should be very careful to ensure that the use of this information and/or software material complies with the laws, rules, and regulations of the jurisdictions with respect to which it is used.

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

U.S. Government Rights. All documents, product and related material provided to the U.S. Government are provided and delivered subject to the commercial license rights and restrictions described in the governing license agreement. All rights not expressly granted therein are reserved.

TeamQuest, the TeamQuest logo, TeamQuest Alert, TeamQuest Analyzer, TeamQuest Baseline, TeamQuest CMIS, TeamQuest Harvest, TeamQuest IT Service Analyzer, TeamQuest IT Service Reporter, TeamQuest Manager, TeamQuest Model, TeamQuest On the Web, TeamQuest Predictor, TeamQuest Online, TeamQuest Surveyor, TeamQuest View, and Performance Surveyor are trademarks or registered trademarks of TeamQuest Corporation in the US and/or other countries.
Unisys and ClearPath are registered trademarks of Unisys Corporation in the US and/or other countries.
All other trademarks and service marks are the property of their respective owners.

About This Guide

TeamQuest Baseline® and TeamQuest Online® are required OS 2200 software packages that provide the services for the TeamQuest Performance Software family. TeamQuest Baseline is installed on a host system and provides the components to collect, store, manage, and administer your performance data.

Purpose

The *OS 2200 TeamQuest Baseline and TeamQuest Online User Guide* describes TeamQuest Baseline and TeamQuest Online. It tells you how to use TeamQuest Baseline as a tool for performing performance management and performance analysis tasks on your system and how to use TeamQuest Online as a performance monitoring tool.

Scope

This guide describes the TeamQuest Baseline and TeamQuest Online software. It provides conceptual information for TeamQuest Baseline and the performance database structure. It tells how to use the probes to collect and store performance data in the performance database.

Audience

This guide is intended for those who are interested in performance analysis and capacity management of OS 2200 systems. TeamQuest Online addresses the needs of the performance analyst. TeamQuest Baseline addresses the needs of the performance analyst and the capacity planner. The performance analyst who is typically interested in detailed data can use TeamQuest Baseline or TeamQuest Online to collect system data at a fine granularity. The capacity planner can use TeamQuest Baseline to collect data across long periods of time. Marketing representatives and technical managers who are interested in the performance management of OS 2200 systems might also use TeamQuest Baseline.

Prerequisites

This guide assumes that you have basic knowledge of the Software Instrumentation Package (SIP), TPM log, IOTRACE, the MAPPER accounting log, the system log, and other existing performance analysis software on the OS 2200 system.

How to Use This Guide

This guide is intended to be used with the *TeamQuest View User Guide* to view TeamQuest Baseline and TeamQuest Online reports. The *OS 2200 TeamQuest Baseline and TeamQuest Online User Guide* contains information specific to the host component of the software. The *TeamQuest View User Guide* describes the workstation software and explains how to use the TeamQuest View® user interface to view reports from data collected by the TeamQuest Performance Software applications.

This guide is divided into two parts. Part 1 contains information that will familiarize you with the TeamQuest Baseline product and the basic concepts needed to use and administer TeamQuest Baseline as installed. It is intended for first-time users and those who have no need to change the default settings. Part 2 contains advanced concepts and information for customizing TeamQuest Baseline to meet the specific needs for your site.

Organization

This guide contains the following:

Part 1. Getting Up and Running

Section 1. Introduction

This section provides an introduction to TeamQuest Performance Software family. It describes the TeamQuest Baseline and TeamQuest Online products and tells how the products work.

Section 2. TeamQuest Baseline and TeamQuest Online Concepts

This section explains the architecture of TeamQuest Baseline and TeamQuest Online and describes the performance database structure.

Section 3. TeamQuest Baseline Administration

This section provides basic information on administering TeamQuest Baseline or TeamQuest Online with the default settings as installed. It contains information on default aggregation sets, and alarm handling. It also outlines steps you can take to ensure that you are collecting the data you need for your site requirements.

Part 2. Customizing Your Site

Section 4. TeamQuest Performance Software Administration

This section provides information to establish and terminate an administration session. TeamQuest Performance Software Administration allows you to perform maintenance tasks for workloads, aggregations, and derived statistics.

Section 5. Customizing Workloads

This section provides information on the workloads used by TeamQuest Baseline. It describes workload characteristics and the definition language for defining workloads. It also tells you how to use the **tqzip** program to unload and load workload definitions to an output file.

Section 6. Customizing Aggregation Sets

This section contains advanced concepts about the aggregation sets in the performance database and tells how you can customize the aggregation sets to meet the particular needs of your site. It provides information for using the **tqasa** and **tqagg** programs to perform tasks for maintaining aggregation sets.

Section 7. Customizing Alarms and Thresholds

This section provides information on the alarms and thresholds capability of TeamQuest Baseline. It tells you how to perform the various tasks in creating and maintaining alarms and thresholds for database parameters and derived statistics.

Section 8. Customizing Derived Statistics

This section contains advanced concepts for using derived statistics with TeamQuest Baseline. It also tells you how to use the **tqzip** program to unload and load your derived statistic definitions to an output file.

Section 9. Customizing Data Collection Probes

This section provides information you need to customize the data collection for each of the TeamQuest probes. It also tells you how to write your own probes for collecting site-specific data.

Section 10. Customizing Performance Databases

This section tells you how create additional databases, delete databases, estimate disk space requirements for a database, clear database locks, and remove database parameters from a database. It also provides backup and recovery information.

Section 11. Customizing Tables

This section provides the information you need to maintain the table data collected by the TeamQuest Baseline probes. It tells you how to use the **tqzip** program to define collection periods and retention specifications, delete records from tables, and register a table in the database. It also tells you how and why to use the **tqrorg** program to reorganize your table files for improved efficiency.

Appendix A. OS 2200 Parameters

This appendix lists the statistics that are collected by the TeamQuest Baseline and TeamQuest Online host software.

Appendix B. Predefined Report Descriptions

This appendix provides a description of each of the predefined reports for your host system.

Appendix C. Report Definition File

This appendix describes the format of the report definition file. It provides the format of the file and a description of the fields specified in the file.

Appendix D. Host-Based Reporting for Aggregation Set Data

This appendix describes how to use the TeamQuest Baseline **tqrpt** program to generate reports containing aggregation set data directly from the performance database on the host system. This program is available with TeamQuest Baseline only.

Appendix E. Host-Based Reporting for Table Data

This appendix describes how to use the TeamQuest Baseline **tqrep** program to generate reports containing table data directly from the performance database on the host system. This program is available with TeamQuest Baseline only.

Appendix F. The tqgetm Program for TeamQuest Model

This appendix describes the **tqgetm** program that allows you to export data from TeamQuest Baseline to TeamQuest Model.

Appendix G. Alarm History Table

This appendix describes the format of the alarm history table.

Appendix H. tqzip Processor Call Format

This appendix serves as a reference for the **tqzip** processor call format.

Appendix I. 2200 Socket Interface

This appendix provides the information necessary to connect an OS 2200 system to TeamQuest View using a socket connection.

Appendix J. 2200 SNMP Trap Alarms

This appendix describes how to define thresholds that will send alarm events to any management console that accepts Simple Network Management Protocol (SNMP) traps.

Results

After reading and using this guide, you will be able to use TeamQuest Baseline or TeamQuest Online on the host system, maintain the performance database, and use the data collection probes to gather measurement data in the database.

Related Product Information

The following related documents may be helpful to you when using TeamQuest Baseline and TeamQuest Online. Refer to the version that corresponds to the level of software in use at your site.

TeamQuest Product Documentation

If you are using TeamQuest manuals, you can find related product information in the following documents.

***TeamQuest Model User Guide for Unisys Systems* (TQ–01424)**

This guide describes how to use the TeamQuest Model® product with Unisys systems. TeamQuest Model allows you to build and solve models to predict the performance of your computer system should conditions change.

***OS 2200 TeamQuest PMLog End Use Reference Manual* (TQ–01351)**

This manual provides information for using the TeamQuest PMLog software with TeamQuest Baseline.

***OS 2200 TeamQuest Probes End Use Reference Manual* (TQ–01361)**

This manual provides information for using the TeamQuest Probes software with TeamQuest Baseline or TeamQuest Online.

***OS 2200 TeamQuest Products Installation Guide* (TQ–01013)**

This guide contains information you will need to install any of the TeamQuest software products for ClearPath OS 2200 systems.

***TeamQuest View User Guide* (TQ–01401)**

This guide describes how to use the TeamQuest View software. TeamQuest View is the user interface to TeamQuest Baseline and TeamQuest Online. You use TeamQuest View to view reports from data collected by the TeamQuest Performance Software applications.

Unisys Product Documentation

If you are using Unisys manuals, you can find related product information in the following documents.

***Business Information Server for ClearPath OS 2200 Administration Guide* (7831 9415)**

This guide provides information system configuration, database management, user and run registration, and configuring communications links with other Business Information Server systems.

***ClearPath OS 2200 Cipher Application Programming Interface (API) Programming Reference Manual* (3826 6110)**

This manual provides specific information on the Cipher API product used for FIPS 140-2 compliant encryption of data. If installed, the user-id and password supplied in the TQTCP configuration element can be encrypted for security reasons for TeamQuest Baseline and TeamQuest Online.

***ClearPath OS 2200 Exec System Software Administration Reference Manual* (7831 0323)**

This manual provides specific information for setting up and administering the OS 2200 system. This information includes instructions for booting, initializing, monitoring, reconfiguring, and recovering the system. This manual also contains information on the Software Instrumentation Package (SIP). TeamQuest Baseline and TeamQuest Online use data collected by SIP for data reduction and report generation.

***ClearPath OS 2200 Software Products Installation Guide* (7831 0612)**

This guide explains how to use the Software Library Administrator (SOLAR) to install OS 2200 software products from a software package tape.

***OS 2200 Exec System Software Operations Reference Manual* (7831 0281)**

This manual explains the operator interface with the system console and peripherals. It also discusses operator responsibilities, describes procedures, and lists keyins.

***OS 2200 Linking System Subsystems Programming Guide* (7830 7451)**

This guide describes software subsystems including how to create, install, and use them. It includes a discussion of security as it relates to the installation and usage of subsystems.

***OS 2200 System Log Operations and Support Reference Manual* (7831 0315)**

This manual explains system log processing and the resulting log entries of the OS 2200 system. The TeamQuest Baseline system log probe collects system log data for its reporting.

***OS 2200 Transaction Processing Administration and Operations Reference Manual* (7830 7881)**

This manual tells you how to turn on the Transaction Processing (TIP) log accounting feature if you have TIP running on your system. It also describes the TIP Performance Monitor (TPM) logging mechanism for reporting Transaction Processing (TIP) and High-Volume Transaction Processing (HVTIP) statistics.

***Single Point Autoaction Message System Administration Guide* (7833 4760)**

This guide provides information on the Single Point Autoaction Message System (SP-AMS). The alarm capability of TeamQuest Baseline can be integrated with the message automation and alarm handling functionality of SP-AMS.

***SMART for ClearPath OS 2200 Administration and Operations Guide* (7830 7337)**

This guide provides information on SMART. The alarm capability of TeamQuest Baseline can be integrated with the message automation and alarm handling functionality of SMART.

How to Access Portable Document Format (PDF) Files

All TeamQuest product documentation is available as PDF files. PDF files can either be downloaded from the TeamQuest Web site or they can be viewed from your product CD or DVD.

You must have the Adobe Reader software from www.adobe.com installed on your workstation to view the PDF files.

To download product documentation from the Web site:

1. Access the TeamQuest Web site at www.teamquest.com.
2. Click the Support tab, then Customer Area.
3. Log in to the Customer Area.
4. Click the Downloads link.
5. Click the Product Documentation link for the desired release level and product platform.
6. Select the document you wish to download.

   When the download is complete, the PDF file will open.
7. Select the Save As command from the File menu to save a copy of this document.

To access product documentation from your product CD or DVD:

1. Load the CD or DVD in your disk drive.

   The disk contents dialog box is displayed.

2. Open the welcome.pdf file.

   The welcome.pdf file can be used to navigate to the desired PDF file.

You may wish to copy the welcome.pdf file and the pdfdocs folder to your server so multiple users have convenient access to the product documentation.

Note: *When copying the welcome file and the pdfdocs directory, you must maintain the established directory structure. If the directory structure is not maintained, the navigation links within the welcome.pdf file will not open the documents located in the pdfdocs directory.*

Notation Conventions

The examples in this guide show, as nearly as possible, how your input, system output, and screens appear on a terminal.

Program names are shown in boldface type.

In this guide, the following conventions apply for command formats when using TeamQuest Baseline and TeamQuest Online on the host system:

- You can enter any commands or other required input in either uppercase or lowercase letters. Both ways are shown in the examples in this guide.
- In command and format examples, anything that appears in regular type must be entered exactly as shown. For example, in the following command:

  `@tqbsp [-r,] -i,`*`parameter-table`*`,`

  you must enter **@tqbsp** exactly as shown.
- Brackets are used to indicate optional entries. In the following command:

  `[-d,`*`databasename`*`,]`

  that which appears between the brackets, -d,*databasename*, is optional. You are not required to enter the -d parameter.
- Italics are used to indicate variable values that you must supply, depending on the circumstances. In the following command:

  `-a,`*`aggset1`*`,`

  you must supply a valid aggregation set name in place of *aggset1*.
- For menu selections within the graphical user interfaces, a command sequence is shown as

  `Administration > Workloads`

  This sequence indicates that to perform the necessary function, you select Workloads from the Administration menu.

Contents

About This Guide

Part 1. Getting Up and Running

Section 1. Introduction

1.1. What Is TeamQuest Performance Software? 1–1
1.2. What Can You Do with TeamQuest Performance Software? 1–3
1.3. What Is TeamQuest Baseline? 1–4
1.4. What Can You Do with TeamQuest Baseline and TeamQuest Online? 1–5
1.5. Processing Overview 1–5
1.5.1. Data Collection 1–6
1.5.2. Data Display 1–6
1.6. How Do TeamQuest Baseline and TeamQuest Online Work? 1–8
1.6.1. The Host Software Component 1–8
1.6.1.1. Probes 1–8
1.6.1.2. Performance Database 1–10
1.6.1.3. Host to TeamQuest View Interface 1–11
1.6.2. TeamQuest View Software 1–11
1.7. Product Compatibility 1–11

Section 2. TeamQuest Baseline and TeamQuest Online Concepts

2.1. Overview 2–1
2.2. Performance Database 2–1
2.2.1. Aggregation Sets 2–1
2.2.2. Parameters 2–2
2.2.3. Data Values 2–3
2.2.4. Table Data 2–4
2.3. Data Collection 2–4
2.4. Workload Management 2–4
2.5. Alarms and Thresholds 2–5
2.6. Derived Statistics 2–6

Section 3. TeamQuest Baseline Administration

3.1. tq-ini Element 3–1
3.2. Aggregation Sets 3–3
3.3. Alarm Handling 3–3
3.3.1. Default Threshold Definitions 3–4
3.3.2. Using the Default Threshold Definitions 3–4
3.3.3. Changing the Default Threshold Definitions 3–5
3.4. What's Next 3–5

Part 2. Customizing Your Site

Section 4. TeamQuest Performance Software Administration

4.1. Introduction 4–1
4.2. What is a Session? 4–1
4.3. Activating TeamQuest Performance Software Administration 4–2
4.4. Creating a Session 4–3
4.4.1. Using a Socket Connection 4–5
4.4.2. Using a SSL Connection 4–6
4.4.3. Using a Named Pipe Connection 4–6
4.5. Additional Sessions 4–6
4.6. Session Timeout Warnings 4–7
4.7. Terminating the Communications Session 4–8

Section 5. Customizing Workloads

5.1. Introduction 5–1
5.2. Concepts 5–1
5.2.1. Workload Qualifiers 5–2
5.2.2. Workload Definition Language 5–2
5.2.2.1. Using Numerical Conditions 5–3
5.2.2.2. Using String Conditions 5–4
5.2.2.3. Using Regular Expressions in String Conditions 5–5
5.2.2.4. Using Boolean Expressions in Definitions . 5–7
5.2.2.5. Definition Example 5–7
5.3. Maintaining Workloads with TeamQuest Performance Software Administration 5–8
5.3.1. Performing Workload Administration Tasks 5–9
5.3.2. Creating a New Workload Set Definition 5–10
5.3.3. Displaying or Modifying a Workload Set Definition 5–11
5.3.4. Deleting a Workload Set 5–12
5.3.5. Creating a New Workload 5–13
5.3.6. Displaying or Modifying a Workload Definition ... 5–16
5.3.7. Importing Workload Sets and Workload Definitions 5–17
5.3.8. Exporting Workload Sets and Workload Definitions 5–17
5.3.9. Deleting a Workload 5–18

5.4. Maintaining Workloads with tqzip 5–18
5.4.1. Unloading Workload Definitions 5–19
5.4.2. Loading Workload Definitions 5–22

Section 6. Customizing Aggregation Sets

6.1. Introduction 6–1
6.2. Concepts 6–1
6.2.1. Aggregation Set Structure 6–2
6.2.2. Moving Window Concept for Data Storage 6–4
6.2.3. Aggregation Set Consolidation 6–4
6.2.4. Feeding Aggregation Sets 6–7
6.2.5. Compact and Expanded Aggregation Set Format 6–7
6.3. Maintaining Aggregation Sets with TeamQuest Performance Software Administration 6–8
6.3.1. Performing Aggregation Set Administration Tasks 6–9
6.3.2. Creating an Aggregation Set 6–10
6.3.3. Displaying or Modifying Aggregation Set Definitions 6–14
6.3.4. Deleting an Aggregation Set 6–15
6.4. Maintaining Aggregation Sets with tqasa 6–15
6.4.1. Creating an Aggregation Set 6–16
6.4.2. Removing an Aggregation Set 6–19
6.4.3. Listing Aggregation Sets 6–20
6.4.4. Unloading Aggregation Set Definitions 6–21
6.4.5. Loading Aggregation Set Definitions 6–24
6.4.6. Converting Expanded Aggregation Sets 6–26
6.4.7. Performing Data Aggregation 6–27

Section 7. Customizing Alarms and Thresholds

7.1. Introduction 7–1
7.2. Concepts 7–1
7.3. Opening the Alarms and Thresholds Menu 7–5
7.4. Creating Threshold Definitions 7–6
7.5. Reusing Threshold Definitions 7–11
7.6. Updating Threshold Definitions 7–12
7.7. Viewing Thresholds 7–13
7.8. Creating Derived Statistic Threshold Definitions 7–14
7.9. Updating Derived Statistic Threshold Definitions 7–16
7.10. Viewing Derived Statistic Threshold Definitions 7–18
7.11. Deleting Thresholds 7–19
7.12. Notifying Probes of Threshold Changes 7–21
7.13. Unloading Threshold Definitions 7–21
7.14. Loading Threshold Definitions 7–24
7.15. Turning Threshold Checking On and Off 7–25
7.15.1. Using the Alarms and Thresholds Menu 7–25
7.15.2. Using the tqzip Processor Call 7–26

7.16. Generating Alarm Lists 7–27
7.17. Using tqalm 7–28
7.18. Alarm Messages 7–30
7.19. Site-Specific Alarm Handling 7–31
7.20. Single Point Operations (SPO) Considerations 7–32

Section 8. Customizing Derived Statistics

8.1. Introduction 8–1
8.2. Concepts 8–1
8.2.1. Derived Statistic Identification 8–1
8.2.2. Parameter References and Substitution 8–2
8.2.3. Formula Definition 8–3
8.3. Maintaining Derived Statistics with TeamQuest Performance Software Administration 8–4
8.3.1. Performing Derived Statistic Administration Tasks 8–5
8.3.2. Derived Statistic Hierarchy 8–6
8.3.3. Creating a New Derived Statistic Definition 8–8
8.3.3.1. Defining an Alias 8–11
8.3.3.2. Qualifying Parameter References 8–12
8.3.3.3. Deleting an Alias 8–13
8.3.4. Saving a Derived Statistic Definition 8–13
8.3.5. Viewing Long Key Names 8–14
8.3.6. Importing Derived Statistic Definitions 8–15
8.3.7. Exporting Derived Statistic Definitions 8–15
8.3.8. Displaying or Modifying a Derived Statistic Definition 8–16
8.3.9. Deleting a Derived Statistic Definition 8–17
8.4. Maintaining Derived Statistics with tqzip 8–17
8.4.1. Unloading Derived Statistic Definitions 8–18
8.4.2. Loading Derived Statistic Definitions 8–21

Section 9. Customizing Data Collection Probes

9.1. Introduction 9–1
9.2. tqctl Runstream 9–1
9.3. SIP Data Collection 9–2
9.3.1. tqbsp Processor Call Statement 9–2
9.3.2. Terminating Data Collection 9–3
9.3.3. Abnormal Termination of Data Collection 9–3
9.3.4. SIP File Input 9–4
9.4. System Activity Monitor Collection 9–5
9.4.1. tqsam Processor Call Statement 9–5
9.4.2. Terminating Data Collection 9–6
9.4.3. Abnormal Termination of Data Collection 9–7
9.4.4. Workload Characteristics for the System Activity Monitor Probe 9–7
9.4.5. tqsam Table Data 9–8
9.4.5.1. Runs Table Information 9–8
9.4.5.2. Transactions Table Information 9–8

9.5. MIPS Metering Data Collection 9–9
9.5.1. tqmips Processor Call Statement 9–9
9.5.2. Terminating Data Collection 9–11
9.5.3. Abnormal Termination of Data Collection 9–11
9.5.4. Trail Selection By Date Range or Relative Time 9–11
9.5.5. System Log File Input 9–12
9.6. Open Distributed Transaction Processing Collection .. 9–14
9.6.1. tqodtp Processor Call Statement 9–14
9.6.2. Terminating Data Collection 9–15
9.6.3. Abnormal Termination of Data Collection 9–15
9.6.4. tqodtp Table Data 9–16
9.6.4.1. System Table Information 9–16
9.6.4.2. Server Table Information 9–16
9.6.4.3. Client Table Information 9–16
9.6.4.4. Service Table Information 9–16
9.7. Real-Time TPM and TPM Log Data Collection 9–17
9.7.1. tqrtpm Processor Call Statement 9–18
9.7.2. Terminating tqrtpm Data Collection 9–19
9.7.3. tqtpm Processor Call Statement 9–20
9.7.4. TPM Log File Input 9–21
9.7.5. Workload Characteristics for the RTPM and TPM Log File Probes 9–23
9.7.6. Log File User Area 9–24
9.8. IOTRACE Data Collection 9–25
9.8.1. Data Collection Runstream 9–25
9.8.2. tqiot Processor Call Statement 9–26
9.8.3. Workload Characteristics for the IOTRACE Probe 9–28
9.9. MAPPER Log Data Collection 9–29
9.9.1. Data Collection Runstream 9–29
9.9.2. tqmpr Processor Call Statement 9–29
9.9.3. MAPPER Log File Input 9–31
9.9.4. Changing Default MAPPER Log File Names 9–32
9.9.5. Workload Characteristics for the MAPPER Log Probe 9–33
9.10. System Log Data Collection 9–33
9.10.1. tqsys Processor Call Statement 9–34
9.10.2. System Log File Input 9–35
9.10.3. Workload Characteristics for the System Log Probe 9–37
9.11. User Data Probes 9–38
9.11.1. Writing a User Probe for Performance Data 9–39
9.11.1.1. Data Storage 9–45
9.11.1.2. Running tqusrprb in Continuous Mode 9–46
9.11.1.3. Installing Your User Probe 9–47
9.11.1.4. Restrictions 9–47
9.11.2. Writing a User Probe for Table Data 9–48
9.11.2.1. Creating a Table 9–49
9.11.2.2. Storing Table Data 9–57
9.11.2.3. Examples of Table Data Collection 9–58
9.11.2.4. Command Syntax 9–62
9.11.2.5. Keyword Definitions 9–65

Section 10. Customizing Performance Databases

10.1. Introduction 10–1
10.2. Performance Databases 10–1
10.3. Database Files 10–2
10.4. Creating Additional Databases 10–3
10.5. Deleting Databases 10–3
10.6. Listing Existing Databases 10–4
10.7. Estimating Disk Space Requirements 10–4
10.8. Listing Database Parameters 10–4
10.9. Removing Database Parameters from a Database 10–6
10.10. Database Locking 10–7
10.11. Backup and Recovery 10–8

Section 11. Customizing Tables

11.1. Introduction 11–1
11.2. Concepts 11–1
11.2.1. Collections 11–1
11.2.2. Retentions 11–2
11.2.3. Registration 11–2
11.2.4. Manual Record Deletion 11–2
11.3. Opening the Table Administration Menu 11–3
11.4. Displaying a List of Tables 11–4
11.5. Administering Collection Periods 11–4
11.5.1. Listing Collection Periods 11–6
11.5.2. Adding a Collection Period 11–7
11.5.3. Deleting a Collection Period 11–8
11.5.4. Turning Data Collection On and Off 11–9
11.5.5. Exiting the Collection Period Menu 11–9
11.6. Defining Retention Specifications 11–9
11.6.1. Listing Retention Specifications 11–11
11.6.2. Modifying the Primary Retention Specification ... 11–12
11.6.3. Adding a Retention Specification 11–13
11.6.4. Deleting a Retention Specification 11–14
11.6.5. Setting the Retention Start Time 11–15
11.6.6. Exiting the Retention Specification Menu 11–15
11.7. Deleting Records from Tables 11–16
11.7.1. Deleting Records Based on Time 11–17
11.7.2. Deleting Records for a System 11–17
11.7.3. Deleting All Records for a Table 11–18
11.8. Registering a Table 11–18
11.9. Deregistering a Table 11–20
11.10. Unloading Table Specifications 11–21
11.11. Loading Table Specifications 11–25
11.12. Reorganizing Table Files 11–26
11.12.1. Why Do I Need To Reorganize My Table Files? . 11–27
11.12.2. Executing tqrorg 11–28

Appendix A. OS 2200 Parameters

A.1. Introduction A–1
A.2. SIP Parameters A–2
A.2.1. System Statistics A–5
A.2.2. Processor Statistics A–7
A.2.2.1. Performance Monitor A–9
A.2.2.2. Dedicated Java Processor Statistics A–14
A.2.3. Memory Statistics A–18
A.2.3.1. Program Load Statistics A–18
A.2.4. Paging Statistics A–22
A.2.5. I/O Activity Statistics A–24
A.2.6. SIP Parameters for the XPC A–36
A.2.7. Queue Statistics A–39
A.3. TPM Log Parameters A–41
A.4. MIPS Parameters A–43
A.4.1. MIPS Probe Derived Statistics A–43
A.5. IOTRACE Parameters A–44
A.6. MAPPER Log Parameters A–45
A.7. System Log Parameters A–46
A.8. Workload Performance Derived Statistics A–51
A.8.1. SIP Probe Derived Statistics A–51
A.8.2. MAPPER Log Probe Derived Statistics A–53
A.8.3. System Log Probe Derived Statistics A–54
A.9. System Activity Monitor Statistics A–58
A.9.1. Runs Table A–58
A.9.2. Transaction Table A–59
A.10. Open Distributed Transaction Processing Statistics A–61
A.10.1. System Table Information A–61
A.10.2. Server Table Information A–62
A.10.3. Client Table Information A–64
A.10.4. Service Table Information A–64

Appendix B. Predefined Report Descriptions

B.1. Introduction B–1
B.2. Report Definition Directory B–1
B.3. SIP Reports B–2
B.3.1. I/O Reports B–2
B.3.2. Java Reports B–6
B.3.3. Memory Reports B–7
B.3.4. Processor Reports B–8
B.3.5. System Reports B–10
B.3.6. Summary Reports B–11
B.4. IOTRACE Reports B–13
B.5. MIPS Report B–14
B.6. MAPPER Log Reports B–14
B.7. System Log Reports B–15
B.8. TPM Log Reports B–17
B.9. Workload Performance Reports B–19

Appendix C. Report Definition File

C.1. Introduction C–1
C.2. Notation Conventions C–2
C.3. File Format C–2
C.4. General Section C–3
C.5. Parameter-Group Sections C–29
C.6. Individual Parameter Setting Sections C–32

Appendix D. Host-Based Reporting for Aggregation Set Data

D.1. Using tqrpt for Condensed Output D–4
D.2. Using tqrpt for Reporting Derived Statistics D–5

Appendix E. Host-Based Reporting for Table Data

E.1. Overview of tqrep E–1
E.2. Command Format E–2
E.3. Report Definition File Format E–3
E.3.1. Report Section Keys E–5
E.3.2. Table Section Keys E–8
E.3.3. Using Special Identifiers in Section Keys E–10
E.4. Using Default Reports E–12
E.5. Exporting Data E–12

Appendix F. The tqgetm Program for TeamQuest Model

Appendix G. Alarm History Table

Appendix H. tqzip Processor Call Format

Appendix I. 2200 Socket Interface

I.1. Communications Application Program Interface I–2
I.2. TQTCP I–2
I.2.1. Background Run I–3
I.2.2. Processor Call I–4
I.2.3. Configuration Element I–4
I.2.3.1. Same User-id For All TQTCP Runs I–5
I.2.3.2. Unique User-ids For Each TQTCP Run I–7
I.2.4. Error Messages I–9
I.2.5. Security Considerations I–10
I.2.5.1. User-id and Password Encryption I–10
I.3. Connecting to the Socket I–11

Appendix J. 2200 SNMP Trap Alarms

J.1. Overview J–1
J.2. CMS 1100 Configuration J–2
J.3. TeamQuest Alarm Administration Interface J–2
J.4. SNMP Trap Contents J–4
J.4.1. SNMP Trap Mappings J–4
J.4.2. Variable Fields J–5
J.5. Management Console Configurations J–6
J.5.1. HP OpenView Configuration J–7
J.5.2. Tivoli Management Environment Configuration .. J–7
J.5.3. Computer Associates Unicenter TNG J–9
J.6. TeamQuest MIB Definition J–9
J.7. Special Considerations J–12

Glossary 1

Bibliography 1

Index 1

Figures

1–1. TeamQuest Performance Software Family 1–2
1–2. TeamQuest Baseline Components 1–7

6–1. Parameter Grouping for an Aggregation Set 6–3
6–2. Data Consolidation from Aggregation Set 1 to Aggregation Set 2 6–6

7–1. Threshold Zones 7–3

E–1. Report Generated by tqrep E–1

Part 1
Getting Up and Running

Section 1
Introduction

1.1. What Is TeamQuest Performance Software?

TeamQuest Performance Software is a family of products which addresses the needs of performance management and performance analysis of your computer systems. This integrated family of products allows you to choose the tools that you need to manage the performance at your site. The TeamQuest Performance Software family consists of the following products:

- TeamQuest Baseline, TeamQuest SMFII, and TeamQuest Manager®

  The TeamQuest Performance Software includes TeamQuest Baseline for Unisys ClearPath OS 2200 systems, TeamQuest SMFII for Unisys MCP systems, TeamQuest Manager for UNIX/Linux systems, and TeamQuest Manager for Microsoft Windows systems. These products provide the services required for all of the products in the TeamQuest Performance Software family. These products provides the following features:

  – performance data measurement, collection, and storage services

  – administrative interfaces

  – alarm handling capabilities

  – basic textual data access and reporting capabilities

- TeamQuest Online

  This product provides a subset of the services provided by TeamQuest Baseline.

All other products in the TeamQuest Performance Software family require the TeamQuest Performance Software.

- TeamQuest Alert®

  This product provides a performance console that allows you to monitor the current health of a large number of systems. TeamQuest Alert obtains performance data and alarm events from TeamQuest Manager on monitored system. It provides a top-down approach to monitoring large numbers of systems with a tightly coupled drill-down capability for problem investigation.

- TeamQuest Model®

  TeamQuest Model is a client application that runs on your workstation. TeamQuest Model provides capacity planning and “what-if” analysis. TeamQuest Model obtains its data from TeamQuest products running on Unisys ClearPath OS 2200 systems, Unisys MCP systems, and UNIX/Linux systems. It provides an easy-to-use interface to a set of powerful modeling algorithms.

- TeamQuest On the Web®

  This product provides performance reports using a browser. TeamQuest On the Web obtains performance data from TeamQuest Manager and displays it using a set of Java applets on a browser. It is very useful for monitoring systems from a remote location and for the automated publication of management performance reports over the Web.

- TeamQuest View®

  This product provides a graphical user interface for detailed data reporting and analysis. TeamQuest View obtains performance data from TeamQuest Baseline, TeamQuest SMFII, or TeamQuest Manager for display and analysis on your workstation. TeamQuest View allows you to monitor the performance of one or more systems and to quickly drill down to the exact cause of a problem.

The TeamQuest Performance Software family of products offers something for anyone who requires access to performance management data. Figure 1–1 shows the relationship between the products within the TeamQuest Performance Software family.

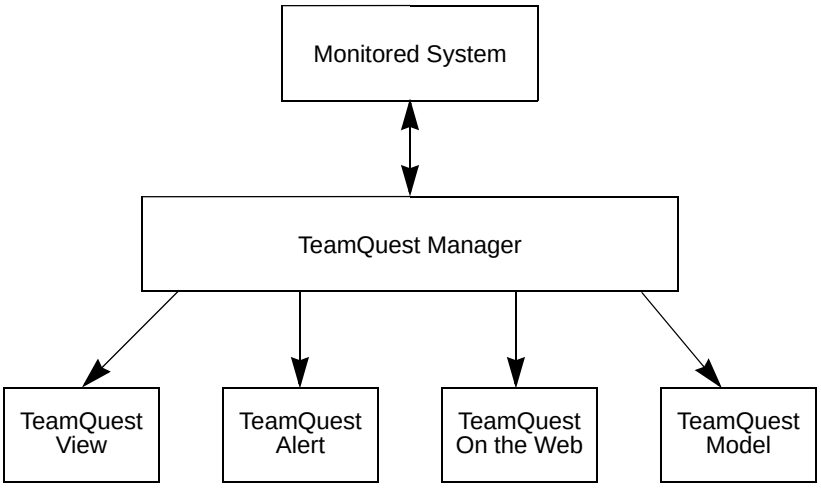


Figure 1–1. TeamQuest Performance Software Family

1.2. What Can You Do with TeamQuest Performance Software?

The TeamQuest Performance Software family allows you to address a wide range of performance problems on your network of computers. When the TeamQuest Performance Software is installed on every monitored system in your network, you can deploy performance applications such as TeamQuest View, TeamQuest Alert, TeamQuest On the Web, and TeamQuest Model, as needed. This structure allows you to deliver the right performance information to the right people at the right time.

The TeamQuest Performance Software allows you to

- Monitor large numbers of systems from a single console. Investigate problems from the same tool with a few clicks of the mouse. Use alarms to detect complex situations and rules of thumb for casual monitoring. Typical users would include: operators and system administrators.
- Monitor one or more systems in detail with graphical and textual reports and use a powerful drill-down facility to investigate problems. Use correlation analysis to detect cause and effect situations. Typical users include: system administrators and performance analysts.
- Analyze historical data with the powerful time selection and reporting capabilities, including trend analysis. Typical users are: performance analysts and capacity planners.
- Characterize workloads along business lines to understand and monitor resource usage and requirements. Typical users are: performance analysts and capacity planners.
- Monitor systems from anywhere you have a standard web browser. Automatically publish current and historical performance reports using your web browser. Typical users are performance analysts (remote or on site), help desk, and management.
- Model your systems for hardware sizing, capacity planning, and what-if tuning scenarios. Typical users include: performance analysts and capacity planners.

1.3. What Is TeamQuest Baseline?

TeamQuest Baseline is the OS 2200 software product that provides the services for the TeamQuest Performance Software family on your OS 2200 system. The TeamQuest Baseline must be installed on every monitored host and consists of the components that collect, store, manage, and administer the performance data.

TeamQuest Baseline performs the following functions for the TeamQuest Performance Software family:

- Data collection and storage

  Data is collected by probes and stored in a performance database. You can select from a predefined set of probes or you can define new probes.

- Administration

  You can control the data that is collected, how often it is collected, and how it is stored. You can also derive statistics and characterize workloads.

- Alarms and Thresholds

  You can specify conditions and actions for the TeamQuest Baseline to test and act upon.

- Application interfaces

  TeamQuest Baseline makes the collected and stored data available to the performance applications (TeamQuest View and TeamQuest Model) using secure application interfaces.

1.4. What Can You Do with TeamQuest Baseline and TeamQuest Online?

The TeamQuest Baseline is the required product within the TeamQuest Performance Software family. TeamQuest Online is a subset of TeamQuest Baseline. These products allow you to perform the following tasks:

- Analyze and view system performance data using a real-time monitor
- Present reports simultaneously from multiple hosts
- Perform correlation analysis between performance statistics
- Produce hard-copy output of reports using a printer or a plotter

TeamQuest Baseline provides the additional capabilities to perform the following tasks:

- Characterize the workloads on the system
- Build and maintain a database containing performance data on global system activity, network activity, and database activity, performance data by workloads, and performance data on process activity
- Perform threshold checking on performance data and derived statistics and generate alarms when values exceed predefined thresholds
- Consolidate data collected at a fine granularity into an aggregation set of coarser granularity
- Collect, reduce, and store process data
- Create your own user data probes
- Generate character-based reports on the host system

1.5. Processing Overview

TeamQuest Baseline and TeamQuest Online processing consists of two phases: data collection and data display. These phases include the various activities involved in creating the performance database, maintaining the database, and using the data stored in the database for report generation and graphics displays. TeamQuest Baseline and TeamQuest Online control the activities for data collection and work with TeamQuest View for data display processing.

1.5.1. Data Collection

TeamQuest Baseline and TeamQuest Online use system monitors in the OS 2200 Executive (Exec) to gather data. TeamQuest Baseline and TeamQuest Online probes collect data from the various system monitors and store that data in the performance database. This data is then available for future retrieval and reporting. The performance database holds all data used to generate reports and to monitor the system in real time. Both TeamQuest Baseline and TeamQuest Online use the performance database for data storage.

1.5.2. Data Display

TeamQuest View, the **tqrpt** program, and the **tqrep** program can be used to display the data collected and stored in the performance database. TeamQuest View is a graphical user interface providing graphical representation of data. The **tqrpt** program provides character-base reporting for aggregation set data (see Appendix D). The **tqrep** program provides character-base reporting for table data (see Appendix E).

During the data display phase, the TeamQuest View software retrieves data from the performance database. You can then use this data to generate reports and display them in textual or graphic format. See the *TeamQuest View User Guide* for more information.

Figure 1–2 shows the components of TeamQuest Baseline. TeamQuest Online consists of all these components except the workload probes (which receive data from HVTIP, TPM, IOTRACE, and MAPPER systems), **tqagg**, **tqalm**, **tqzip**, **tqrep**, and **tqrpt**.

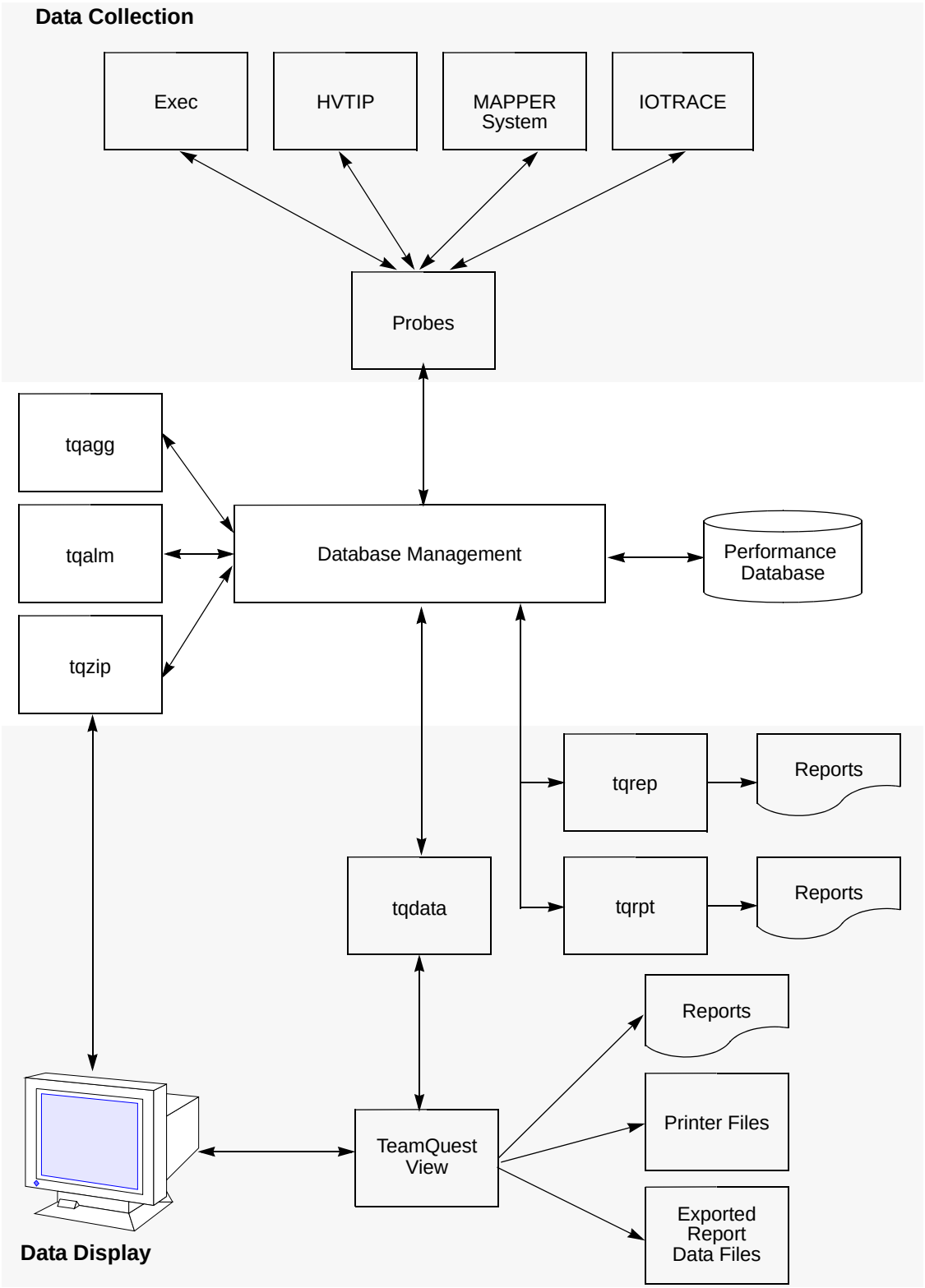


Figure 1–2. TeamQuest Baseline Components

1.6. How Do TeamQuest Baseline and TeamQuest Online Work?

TeamQuest Baseline and TeamQuest Online consist of a host software component and the TeamQuest View user interface software. The host software collects performance data, stores this data in the performance database, and maintains the database. TeamQuest View requests performance data from the performance database and creates reports.

1.6.1. The Host Software Component

The TeamQuest Baseline and TeamQuest Online host software runs on your OS 2200 system collecting data and maintaining the performance database. The host software uses probes to collect data. Individual probes collect data from different sources. Once the data is collected, TeamQuest Baseline or TeamQuest Online stores the data for future retrieval.

1.6.1.1. Probes

The TeamQuest Baseline data collection probes are initiated on the host system. These probes start the data collection and automatically process the data. They also extract statistical values and date and time information for performance parameters and store that information in the performance database.

TeamQuest Baseline and TeamQuest Online come packaged with the **tqbsp** probe. The **tqbsp** probe collects Software Instrumentation Package (SIP) data from the Exec. Using the ER SYSBAL$ mechanism, the **tqbsp** probe starts the data collection and automatically processes that data. The **tqbsp** probe extracts statistical values for performance parameters and stores this information in the performance database.

TeamQuest Baseline comes packaged with the following additional probes that provide the workload capability of the product:

- **tqsam**

  The **tqsam** probe collects information on active runs and transactions using calls to Exec INSP$. TeamQuest Baseline contributes additional capability to dynamically put the data collected by the **tqsam** probe into workloads and to change various collection attributes.

- **tqrtpm**

  The **tqrtpm** probe processes TIP/HVTIP statistics from the TIP Performance Monitor (TPM) audit trails created when TIP logging is configured. This probe requires that your TPM audit trail be configured to mass storage or moved to mass storage prior to starting either probe. The **tqrtpm** probe runs continuously to report the data in near real time mode.

- **tqtpm**

  The **tqtpm** probe processes TIP/HVTIP statistics from the TIP Performance Monitor (TPM) audit trails created when TIP logging is configured. This probe requires that your TPM audit trail be configured to mass storage or moved to mass storage prior to starting either probe. The **tqtpm** probe automatically processes the TPM audit trail(s) from the previous day, from TPM audit trail(s) specified on the call statement, or based on a relative time selection also specified on the call line. The **tqtpm** probe extracts statistical values plus date and time information from the TPM audit trail(s), determines the workload that used the resource, and stores this information in the performance database according to the workloads defined for the system.

- **tqiot**

  The **tqiot** probe processes statistics from the IOTRACE data collection file. The **tqiot** probe processes the IOTRACE data files specified on its processor call statement. The **tqiot** probe extracts statistical values on system resource utilization plus date and time information from the data file, determines the workload that used the resource, and stores this information in the performance database according to the workloads defined for the system.

- **tqmpr**

  The **tqmpr** probe processes data from the MAPPER accounting log files. The **tqmpr** probe processes the set of MAPPER accounting log files from the previous day or the MAPPER accounting log files specified on its processor call statement. **tqmpr** extracts statistical values on MAPPER resource utilization plus date and time information from the log file, determines which workload used the resource, and stores this information in the performance database according to the workloads defined for the system and the MAPPER environments.

- **tqsys**

  The **tqsys** probe processes data from the system log files. The **tqsys** probe automatically processes the system log files from the previous day or the system log files specified on its processor call statement. The **tqsys** probe extracts statistical values on system resource utilization plus date and time information from the log file, determines the workload that used the resource, and stores this information in the performance database according to the workloads defined for the system.

- **tqodtp**

  The **tqodtp** probe collects information in an Open Distributed Transaction Processing (ODTP) environment. The **tqodtp** probe retrieves information from an ODTP environment for configured clients, servers, and system service programs.

TeamQuest Baseline provides two user probe interface programs, called **tqusrprb** and **tqtblprb**, that allow you to develop your own site-specific probes. These programs, coupled with user-supplied information, insert data into the performance database. Any number of user-developed probes can exist on your system.

The TeamQuest Probes software is also available for use with TeamQuest Baseline and TeamQuest Online. This separately packaged product contains additional probes to gather and store data from other system monitors. It provides probes for the Message Control Bank (MCB), the Universal Data System (UDS), and mass storage. See the *OS 2200 TeamQuest Probes End Use Reference Manual* for more information.

The TeamQuest PMLog software provides an additional probe for the TeamQuest Baseline software. It gathers ClearPath OS 2200 Performance Monitor log data for TeamQuest Baseline. See the *OS 2200 TeamQuest PMLog End Use Reference Manual* for more information.

1.6.1.2. Performance Database

The performance database holds all data used to generate history reports and to monitor the system in real time. Data is stored into various aggregation sets in the database. An aggregation set defines the level of granularity of the data and the length of time data is stored.

The performance database, which is maintained automatically on a daily basis, is created during the initial product installation and set-up process. Additional databases can be created.

1.6.1.3. Host to TeamQuest View Interface

The host to TeamQuest View interface is called **tqdata**. You use the TeamQuest View user interface software to connect to the host system using a socket interface. A run is automatically started on the host system which executes **tqdata** and handles all communications between the host and TeamQuest View.

1.6.2. TeamQuest View Software

The TeamQuest View software provides the user interface to the TeamQuest Baseline and TeamQuest Online host software. TeamQuest View lets you access the host system, retrieve data, and create reports.

Using TeamQuest View, you can generate reports showing historical data and view system performance data by using the real-time monitor. Each of the probes is released with a set of predefined reports. The SIP reports give a general overview of system performance on your system. The TPM reports show high-volume TIP/HVTIP usage by workload. The IOTRACE probe reports I/O information by workload. The MAPPER log reports show MAPPER accounting log data by workload. The system log reports show the usage of system resources by workload.

1.7. Product Compatibility

When using TeamQuest Probes or TeamQuest PMLog with TeamQuest Baseline or TeamQuest Online, you must ensure that the base levels of the products are the same. The result of mixing base product levels could be database corruption. For example, you can run a level 7R3 product level with a level 7R3A product, but it is incorrect to run a level 7R2 product with a level 7R3 or a level 7R3A product.

Section 2
TeamQuest Baseline and TeamQuest Online Concepts

2.1. Overview

The TeamQuest Baseline consists of the components that collect, store, manage, and administer the performance data. This section describes the major concepts involved.

2.2. Performance Database

The TeamQuest Baseline probes obtain performance data from the server system and store the data in the performance database. The performance database consists of data values for the parameters gathered from the various collection sources. Data is collected and stored in the database according to defined data aggregation sets. Aggregation sets, parameters, and data values are concepts essential to the understanding of the performance database and the use of TeamQuest View to retrieve and display data from the database.

Table data is also maintained in the performance database at the interval where it was originally collected. Like aggregation set data, table data can be retrieved and displayed by using the TeamQuest View.

2.2.1. Aggregation Sets

The TeamQuest Baseline lets you collect and store your performance data at different sampling intervals or levels of granularity. Depending on your needs, you may be interested in data sampled at a small granularity, such as 1-minute intervals, or at a larger granularity, such as 1-hour intervals. TeamQuest Baseline and TeamQuest Online use aggregation sets to define the granularity of the data in the database, as well as the length of time the data is retained in the database.

TeamQuest Online allows only one aggregation set in the database. Data records are saved into a 1-minute aggregation set. This means that data is collected at 1-minute intervals and is retained in the database for a period of 32 hours.

TeamQuest Baseline allows multiple aggregation sets in your database. So, for example, if you are interested in real-time monitoring of your system, you might have an aggregation set that collects and stores data at 30-second intervals and retains that data for a 1-day time period. If you are interested in trend analysis of your system, you might want to store data at half-hour intervals for a 6-month period. You can have both of these aggregation sets in your database, as well as others that you defined to meet your particular needs. TeamQuest Baseline is released with some default aggregation sets (see 3.2). You can also define your own aggregation sets by using TeamQuest Performance Software Administration (see 6.3).

2.2.2. Parameters

The term *parameter* refers to a performance data item in the database. A parameter consists of a statistic and optional resource, workload, and application group qualifiers. The term *parameter* applies only to performance data stored in the performance database aggregation sets; it does not apply to table data.

A statistic describes the information being collected and stored in the database. A statistic is defined by a hierarchy of key names, which identify the statistic for retrieval from the database. The key names used for the classification of statistics are system, category group, category, subcategory, and statistic name.

- System specifies the system on which the data was collected. A system is defined as a group of closely coupled components that function together and allow measurement of system performance characteristics.
- Category group identifies a group of categories with similar characteristics. Any given system may have many categories and the category group specification is used to group related categories. For example, category groups can include processor, memory, IO activity, and system.
- A category breaks down a category group. Examples of categories within the category group memory are utilization, dynamic allocator, and program load.
- A subcategory further breaks down a category into a group of related statistics. For example, a category called processor can be grouped into subcategories of Exec requests, utilization, and hardware interrupts.
- A statistic name is the name of the information being collected and stored in the database. For example, percent busy, requests per second, and K words per request are statistic names.

The following examples show the hierarchy of key names for various statistics:

System	Category Group	Category	Subcategory	Statistic Name
CL01	Processor	Total	Utilization	Total Exec
CL01	IO Activity	Logical Channel	Cache Summary	Transfer Time
CL01	Memory	Paging Activity	Working Sets	Inputs/Sec

Resource Specification

A parameter can be further qualified by a resource specification. A resource is a component of the system that does work. A resource could be a physical resource, such as a disk unit or tape unit. Or, it could be a logical resource, such as a system call. For example, a statistic with a hierarchy of CL01, IO Activity, Logical Channel, Cache Summary, Transfer Time can be further qualified by the specific logical channel of LC1. In this case, TeamQuest Baseline or TeamQuest Online would report the transfer time statistics for LC1 of your system.

Workload Specification

With the workload capability of TeamQuest Baseline, a parameter can be further qualified by a workload specification. A workload is a logical classification of the work performed on the system. These classifications are site-specific and are expressed in terms that are meaningful to the business. For example, workloads can be defined to track how much of the system resources are used by the payroll department, the order entry department, or a software development project.

Note: *The workload capability is not available with TeamQuest Online.*

Application Group Specification

A parameter can further be qualified by an application group specification. For the MAPPER probe (**tqmpr**), the MAPPER name is specified using the application group specification. For the UDS and MCB probes supplied with the TeamQuest Probes product, the UDS or MCB application name is specified using the application group specification. For parameters collected by any of the other TeamQuest Baseline probes, the application group specification is null.

2.2.3. Data Values

The term *data value* refers to a specific instance of data in the database for a specific parameter. A data value is identified by the parameter, aggregation set, and time and date. Based on this, when you request values from the database, you are requesting statistics for a particular time and date that were stored at a specified granularity.

Note: *Data for an interval t to t+1 is stored at time t. Therefore, when you see a data value at 10:00 a.m., it represents the activity for the interval starting at 10:00 a.m. In the case of snapshot values such as disk space available, the data value at 10:00 a.m. represents the disk space available at the end of the interval started at 10:00 a.m.*

2.2.4. Table Data

The term *table data* refers to the data that is stored in tables. A table is a collection of records containing columns that are related to one another in some way. For example, all data about an alarm is stored in a table. A record contains one set of column values for the table. For example, data for a particular alarm on a system at a given time makes up one record in the alarm table.

Each field in a table contains a particular type of data. The field may contain a string or a number. The numbers may be integers, real numbers, elapsed times, or timestamps. Selected fields in a table can be used to help identify records in the table. These fields are called identifiers. Examples of identifiers are system name, severity, and alarm identifier.

Tables are created by the TeamQuest Baseline probes. Table records are inserted by TeamQuest Baseline and the TeamQuest Baseline probes as is appropriate for the type of records being created.

2.3. Data Collection

TeamQuest Baseline and TeamQuest Online use probes to gather data. The probes collect data about the host system on which they are running. They also extract statistical values, and date and time information for performance parameters, and store that information in the performance database.

2.4. Workload Management

Note: *The workload capability is not available with TeamQuest Online.*

TeamQuest Baseline allows you to track the activity on your system based on a logical classification of the work performed on the system. This logical classification of work is called workload characterization. Workload characterization is the process whereby groups of related performance data are organized into meaningful units called workloads. You define workloads for your performance database and the TeamQuest Baseline probes distribute the data they collect according to the defined workloads. The workload data is stored in the aggregation sets of the performance database.

Workload characterization is used to summarize a large set of data in ways that reduce the level of detail required to understand the operation of the computer system and its relationship to the various business operations it supports. This process is essential to understanding how well your computer system is meeting the requirement of providing acceptable levels of service to the users.

Workload characterization allows you to identify the workloads and track their use of system resources. Each workload is uniquely identified and defines a criteria for distributing the system resources used by each activity. For example, you can define your workload to track how much of the system resources are used by payroll, order entry, or a software development project. The system activity monitor, TPM, IOTRACE, MAPPER log, and system log probes use the workload definitions to distribute the statistics for the work performed by each activity to the appropriate workload.

TeamQuest Baseline supports the definition of multiple workload sets, each containing unique workload definitions. A workload set is defined as a set of workloads that represents the system resources used by all activities on the system. TeamQuest Baseline uses each workload set for distributing the work performed by each activity. This allows you to track the activities on the system using a different view of the system. Creating multiple workload sets is useful for testing new workload definitions because they can be used without affecting the existing production workloads.

TeamQuest Performance Software Administration and the host-based **tqzip** program are used to perform workload management tasks on your system. Workload statistics can be viewed using the TeamQuest View application.

2.5. Alarms and Thresholds

Note: *The alarms and thresholds capability is not available with TeamQuest Online.*

The TeamQuest Baseline probes collect data from various data sources at the sample interval defined for the user-specified aggregation set. The probes can also check the data obtained at each sample interval to determine if the data values exceed user-specified threshold limits. If a data value for a parameter is found to exceed a threshold, the probe can generate an alarm to alert you that a threshold condition has been met. The **tqalm** program handles all alarms generated by the TeamQuest Baseline probes. **tqalm** sends console messages, makes the alarm messages available for display by TeamQuest View, and performs site-specific actions for each alarm. The **tqalm** program can optionally send an SNMP trap when an alarm occurs (see Appendix J). The **tqalm** program also performs threshold checking on any derived statistics for which thresholds are defined.

2.6. Derived Statistics

Note: *The derived statistic capability is not available with TeamQuest Online.*

TeamQuest Baseline allows you to create your own statistics that are derived using data values collected by the TeamQuest Baseline data collection probes. You create derived statistics using TeamQuest View. Once you have created a derived statistic, it can be used in history, monitor, correlation, and host-based reporting. You can also use the **tqzip** program to define thresholds for your derived statistics. The thresholds are used by **tqalm** to determine when alarm messages need to be generated for a derived statistic.

A derived statistic is saved in the performance database as a formula. The actual data values reported for a derived statistic are not stored in the database, but are calculated at report time. This has the following advantages:

- You can report on a statistic that is not being collected by the existing probes.
- You can report on a derived statistic for any time period that contains the data needed to calculate its value. In a sense, the derived statistic exists from the first day you collect data.
- You can modify a derived statistic formula and report on the same time period as before; however, this time you see data values reported that are based on the new formula.
- Because the data values are not stored, minimal mass storage space is required for derived statistics.

No standard deviation is available for reporting with derived statistics. Also, no data is reported for a derived statistic if the formula could not be solved or if any of the components of the formula had no data for the time period selected. Any required data consolidation is done on each of the components of the formula before solving the formula.

Section 3
TeamQuest Baseline Administration

3.1. tq-ini Element

TeamQuest Baseline, TeamQuest Online, TeamQuest PMLog, and TeamQuest Probes use a tq-ini element to obtain execution defaults. The tq-ini element is contained in the installation file for each of the products. These files are

TeamQuest Baseline	SYSLIB*TQ-BASELINE.
TeamQuest Online	SYSLIB*TQ-ONLINE.
TeamQuest PMLog	SYSLIB*TQ-PMLOG.
TeamQuest Probes	SYSLIB*TQ-PROBES.

The General, Database, and Aggregation Sets sections of the tq-ini element contain settings that can be updated as needed. Do not update any of the settings in the other sections of the tq-ini element. When updating the tq-ini element, be sure to update it in all of the product installation files that apply to your site.

Example tq-ini element

The following shows a sample tq-ini element.

```
[General]
;
;  ---------------- (Start of Unix-specific settings) ----------------
;  Log Directory specifies the directory where the log files should be
;  stored.  A relative or absolute path name can be given. Absolute
;  path names must start with a /.  Relative path names are relative
;  to the parent directory of the executable. Be sure there is no
;  white space at the end of the line.
;
Log Directory=log
;
;  Optional Probes specifies optional probes to be controlled by tqdaemon.
;
;Optional Probes=tqsyb tqora
;
;  ----------------- (End of Unix-specific settings) -----------------
;
;  Locale specifies the language setting to be used at runtime.
;  It can be either C (for English) or a platform-specific identifier
;  for Japanese.
;
;  For 2200 Series systems:
;     Use ja to specify Japanese as the locale.
```

```
;
;  For Unix systems:
;     Use the same value that is specified on the LANG environment
;     variable setting for that platform (eg., ja, japan, or japanese).
;
;  Changing the setting to specify Japanese enables support for multibyte
;  characters and causes messages to be output in Japanese.
;
Locale=C

[Database]
;
;  For 2200 Series systems:
;     Data Directory specifies the qualifier to be used for all
;     database files.  A dollar sign ($) is appended to Data Directory
;     to make the actual qualifier (eg., tqdata$).
;
;  For Unix systems:
;     Data Directory specifies the directory where the database files
;     should be stored.  A relative or absolute path name can be
;     given.  Absolute path names must start with a /. Relative path
;     names are relative to the parent directory of the executable.
;     Be sure there is no white space at the end of the line.
;
Data Directory=tqdata
;
;  Database specifies the default database to be used by all of the
;  executables.
;
Database=Production

[Aggregation Sets]
;
;  Aggset1 & Aggset2 specify the aggregation sets that
;  are to be directly populated by the probes.  There can be one
;  or two aggregation sets directly populated by probes. Other
;  aggregation sets must be fed from those aggregation sets and
;  filled by tqagg.
;
Aggset1=1-minute
Aggset2=10-minute
```

3.2. Aggregation Sets

The TeamQuest Baseline capability to support multiple aggregation sets in a single database provides a way for consolidating data stored at a fine sampling rate. By default, the performance database released with TeamQuest Baseline has four aggregation sets. The first aggregation set stores data sampled at 1-minute intervals for a period of 8 hours. The second aggregation set holds data sampled at 10-minute intervals for a period of 8 days. The third aggregation set holds data sampled at 1-hour intervals for a period of 5 weeks. The fourth aggregation set holds data sampled at 8-hour intervals for a period of 13 months.

By default, the TeamQuest Baseline probes store data into the 1-minute and the 10-minute aggregation sets. The aggregation process (carried out by the **tqagg** program) in turn consolidates the 10-minute data into the 1-hour aggregation set and the consolidated 1-hour data into the 8-hour aggregation set. See Section 6 for information on customizing the aggregation sets for your data collection.

Notes:

- *All of the default aggregation sets are in compact format. If you prefer expanded aggregation sets, you must use either TeamQuest Performance Software Administration or the* ***tqasa*** *program to delete and re-create the aggregation sets in expanded format. See 6.2.5 for information on compact and expanded aggregation set format.*
- *The workload probes (****tqtpm****,* ***tqrtpm****,* ***tqiot****,* ***tqmpr****,* ***tqsys****, and* ***tqtlp****) only store data into the one aggregation set. The name of the aggregation set must be specified on the processor call statement using the -a parameter. The supplied runstreams specify the 10-minute aggregation set on the processor call statements.*
- *TeamQuest Online has only one default aggregation set defined. That aggregation set collects data at 1-minute intervals and maintains the data in the performance database for 32 hours.*

3.3. Alarm Handling

The TeamQuest Baseline probes perform threshold checking for all parameters and derived statistics that have a threshold definition in the performance database. When an alarm condition is encountered, the probe generates an alarm. Alarms are handled by the **tqalm** program that is provided with TeamQuest Baseline. By default, the **tqalm** program sends alarm messages to the system console.

See Section 7 for information on how to set up and maintain your threshold definitions.

3.3.1. Default Threshold Definitions

A set of default thresholds is included with TeamQuest Baseline. To use these default thresholds, you must first load the threshold definitions into your performance database. The element SYSLIB*TQ-BASELINE.THRESHOLDS contains the threshold definitions. The following threshold definitions are included:

Alarm ID	Description
Busy IPs	The IP has been fully utilized for a sustained period of time.
Busy Disk	Disk utilization has been high for a sustained period of time.
Low Free Real Memory	The number of free real memory pages is low.

3.3.2. Using the Default Threshold Definitions

After installing and setting up TeamQuest Baseline on your system, you can initiate threshold checking using the default threshold definitions. Perform these steps:

1. Ensure that the TeamQuest Baseline data collection probes and the **tqalm** program are running on your system. If they are not yet running, you can start them by entering:

   ```
   @start sys$lib$*run$.tqctl
   ```

2. Turn on alarm checking.

   ```
   @tqzip -a,on
   ```

3. Load the threshold definitions into your database.

   ```
   @tqzip -l,-e,-t,sys$lib$*tq-baseline.thresholds
   ```

Note: *When you use the -e option with the -l option on **tqzip**, the threshold definitions are loaded without first deleting any existing definitions from your database. If you want to remove all existing threshold definitions from your database before loading the new ones, use **tqzip** with the -l option, but without the -e option.*

3.3.3. Changing the Default Threshold Definitions

If you want to change any of the default threshold information in SYSLIB*TQ-BASELINE.THRESHOLDS, you can use a text editor to update the thresholds element before loading it into the database. See 7.3 for more information on the threshold definition file format. This section contains information you need to consider when updating the threshold element. Threshold definitions can be updated and reloaded as many times as necessary. Each time you load thresholds with **tqzip** the probes automatically begin using your latest threshold changes.

You can also use the **tqzip** interactive interface to maintain your threshold definitions. See Section 7 for additional information on how to set up alarms and thresholds for your system.

3.4. What's Next

The following is a brief overview of the recommended steps that you can use to verify that the correct data is being collected after completing the basic installation of TeamQuest Baseline.

1. Analyze the aggregation sets to ensure data is being collected in meaningful intervals. By default, TeamQuest Baseline collects data at 1-minute and 10-minute intervals. These two aggregation sets feed the 1-hour and 8-hour aggregation sets for capacity planning and trend analysis. Systems that are being monitored for performance purposes may require a longer data retention period for the 10-minute aggregation set, while systems being monitored for capacity planning may require data retention longer than the 13-month default for the 8-hour aggregation set. You can configure your aggregation sets for your site-specific requirements. See Section 6 for more information on maintaining the aggregation sets of your database.
2. Configure workload definitions. Workload definitions may be a consideration at your site. Workloads provide the capability to group user system utilization. This is meaningful in determining usage by each group (for example, development 20%, administration 15%, database 58%, and other 7%). Workloads provide a method of determining the resource utilization for a given group of users. See Section 5 for information on maintaining the workloads for TeamQuest Baseline.
3. Configure alarms and thresholds to send alerts to the console based on user-defined thresholds for system parameters. This is very helpful in monitoring systems that require minimal service, as well as systems operating with restrictive limits. Alarms may send console messages and call a user-supplied routine for site-specific alarm handling. See Section 7 for more information about defining alarms and thresholds for your system.
4. Create derived statistics. Derived statistics are user-defined formulas made up of collected parameters. For example, the parameters collected from the system may not provide an overall average statistic and by using a combination of collected statistics, you can define a derived statistic to represent the average. Derived Statistics are stored as formulas in the database and are calculated at report time. Derived statistics can be added to any report or included on a user-defined report. See Section 8 for more information on derived statistics.

Part 2
Customizing Your Site

Section 4
TeamQuest Performance Software Administration

4.1. Introduction

TeamQuest Performance Software Administration allows you to perform the maintenance functions for workloads, aggregations, and derived statistics through a graphical user interface.

4.2. What is a Session?

Before performing any maintenance tasks using TeamQuest Performance Software Administration, a connection must be made to the system where the data is being stored. This connection to the data is called a session.

A session defines how you connect to the host where the performance database exists. A session definition identifies the name of the particular performance database you are using. Once all this information is obtained, it is saved and identified by a single name, which can then be used on subsequent queries of that data.

Each session that is opened requires a separate communications connection. The total number of sessions that can be opened at the same time is limited to a maximum of 15 and may be further limited by the available memory on your workstation. Multiple session connections to the same host may be restricted by the number of connections available.

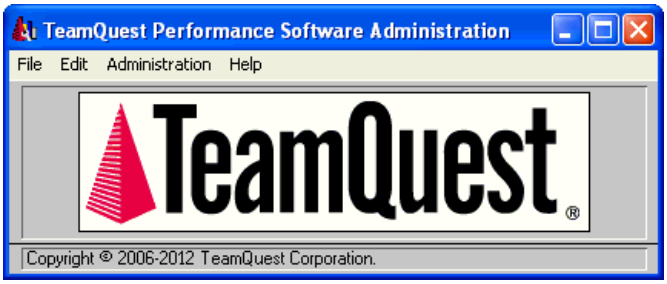
When you start TeamQuest Performance Software Administration, you immediately encounter sessions. The interface will prompt you to open a new session.

4.3. Activating TeamQuest Performance Software Administration

When TeamQuest Performance Software Administration has been correctly installed on your system, you can activate the interface from the Start menu.

Select Start > Programs > TeamQuest > TeamQuest Performance Software Administration. TeamQuest Performance Software Administration is activated and you and you are prompted to create a session (see 4.4).

Once the session is established, the TeamQuest Performance Software Administration window is displayed.



All administration tasks are performed using the Administration menu in this window. TeamQuest Performance Software Administration allows you to perform the following tasks:

- Maintaining Workloads with TeamQuest Performance Software Administration (see 5.3)
- Maintaining Aggregation Sets with TeamQuest Performance Software Administration (see 6.3)
- Maintaining Derived Statistics with TeamQuest Performance Software Administration (see 8.3)

4.4. Creating a Session

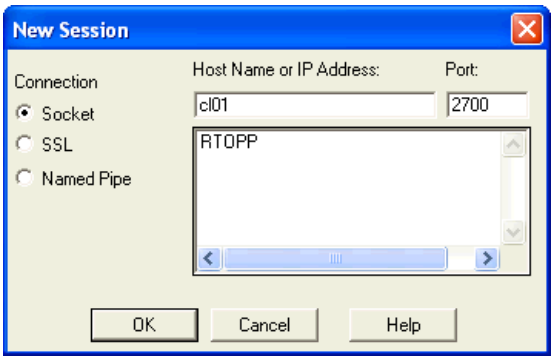
A session defines how the TeamQuest Performance Software Administration connects to the host where a TeamQuest performance database exists. The session information also identifies the name of the performance database you are using. By creating a session definition and saving this information under a session name, you can reuse it for subsequent queries of that data.

When you create a new session, you will specify how to connect to the host software where your performance database exists. You can connect to the host software using a socket or named pipe.

To create a new session:

1. Select File > New Session.

   The New Session dialog box is displayed. This dialog box is used to select the type of communications connection to access the performance database.



2. From the Connection option buttons, select the type of connection you want to use for the session you are defining.

   When you select the appropriate option button for your connection, the fields on the right side of the dialog box will change to allow you to provide the information required to make the connection.

   Socket—For more information, see 4.4.1.

   SSL—For more information, see 4.4.2.

   Named Pipe—For more information, see 4.4.3.

3. Enter the information required to make the communications connection.

4. Click the OK button.

   The New Session dialog box closes and TeamQuest Performance Software Administration initializes communication with the host software. Messages indicating the progress of the initialization are displayed in the status bar at the bottom of the main window.

   Each time you define a new session, the host name or IP address is added to the list of currently defined hosts.

New Session Dialog Box

The New Session dialog box contains the following fields:

Connection
: These option buttons allow you to select the type of connection to be used to access the performance database. The fields on the right side of the dialog box will change depending on the type of connection you select in this field.

Socket Settings
: These fields are used to specify the settings for socket connections.

  Host Name or IP Address
  : This field contains the name of the host or the IP address of the system where the performance database resides.

  Port
  : This field contains the port identification for the socket to connect to.

  Socket name list
  : This list box contains a list of the socket names that have already been used with a session. Each time you define a new session, the host name or IP address is added to this list.

  For more information, see 4.4.1.

SSL Settings
: These fields are used to specify the settings for SSL connections.

  Host Name or IP Address
  : This field contains the name of the host or the IP address of the system where the performance database resides.

  Port
  : This field contains the port identification for the socket to connect to.

  Socket name list
  : This list box contains a list of the socket names that have already been used with a session. Each time you define a new session, the host name or IP address is added to this list.

  For more information, see 4.4.2.

Named Pipe Settings

These fields are used to specify the settings for named pipe connections.

Host Name or IP Address

This field contains the name of the host or the IP address of the system where the performance database resides.

Named Pipe list

This list box contains a list of the names of the named pipes that have already been used with a session. Each time you define a new session, the host name or IP address is added to this list.

For more information, see 4.4.3.

4.4.1. Using a Socket Connection

When you select the Socket option button from the New Session dialog box, you need to specify a host name or an Internet Protocol (IP) address and a port number. You can either enter a host name or IP address in the edit box or you can choose from the current list of hosts (below the edit box) that are already defined as part of a session. You may need to specify the full domain name of the host to which you want to connect. Each time you define a new session, the host name or IP address is added to the list of currently defined hosts.

Note: *Before using sockets you must make sure your host software is configured to accept socket connections on that host. See Appendix I for more information on socket connections.*

The port number defaults to the value 2700.

If you need to remove any host names or IP addresses, select the item you wish to remove from the list and press your keyboard Delete key.

4.4.2. Using a SSL Connection

When you select the SSL option button from the New Session dialog box, you need to specify a host name or an Internet Protocol (IP) address and a port number. You can either enter a host name or IP address in the edit box or you can choose from the current list of hosts (below the edit box) that are already defined as part of a session. You may need to specify the full domain name of the host to which you want to connect. Each time you define a new session, the host name or IP address is added to the list of currently defined hosts.

Note: *Before using sockets you must make sure your host software is configured to accept socket connections on that host. See Appendix I for more information on socket connections.*

The port number defaults to the value 2703.

If you need to remove any host names or IP addresses, select the item you wish to remove from the list and press your keyboard Delete key.

4.4.3. Using a Named Pipe Connection

When you select the Named Pipe option button from the New Session dialog box, you need to specify a host name or IP address. You can either enter a host name or IP address in the edit box or you can choose from the current list of hosts (below the edit box) which are already defined as part of a session. You may need to specify the full domain name of the host to which you want to connect. Each time you define a new session, the host name or IP address is added to the list of currently defined hosts.

Note: *Named Pipe connections are supported for connecting to Unisys systems only.*

If you need to remove host names or IP addresses, select the item you wish to remove from the list and press your keyboard Delete key.

4.5. Additional Sessions

If you have multiple hosts collecting data, you can define and open additional host communications sessions to maintain your workloads, aggregations, and derived statistics.

To open additional sessions, select File > New Session and follow the instructions for creating a new session (see 4.4).

4.6. Session Timeout Warnings

Warning screens may appear during your administration session. When a request is made to the host and the host does not acknowledge the request within the specified timeout period, TeamQuest Performance Software Administration displays the Session Timeout window.

The following two messages can be displayed in the Session Timeout window:

Session has not acknowledged request.
: Host database server has still not acknowledged a data request sent to it within the sessions communications timeout time.

Session sent an acknowledgement.
: The host database server had acknowledged a data request sent to it but not within the sessions communications timeout time.

If the time listed in the Session Timeout window is past the current time on the host, your communications session may have been destroyed due to a failure on the host. You should close the session by selecting File > Terminate Session.

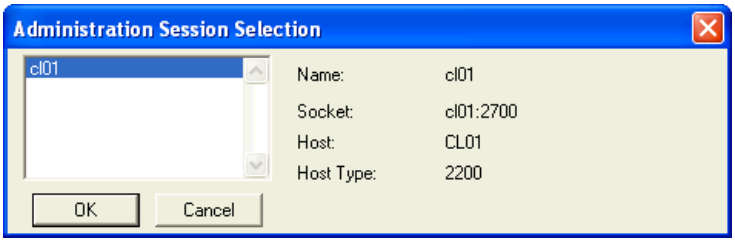
If the host replies to the outstanding data request, TeamQuest Performance Software Administration updates the Session Timeout window. TeamQuest Performance Software Administration automatically updates the Current Timeout Value to reflect the amount of time the host took to respond to the message. TeamQuest Performance Software Administration leaves this window open to let you know that communications on the session has slowed down.

4.7. Terminating the Communications Session

To terminate a session:

1. Select File > Terminate Session.

   The Administration Session Selection dialog box is displayed.



2. From the list of open sessions, select the session you want to terminate.

   The session that you select is described in the session information on the right side of the dialog box.

3. Click OK.

   The selected session is terminated and you are returned to the main window.

When you have completed your administration tasks and terminated the open sessions, close the TeamQuest Performance Software Administration window by selecting File > Exit.

Section 5
Customizing Workloads

5.1. Introduction

TeamQuest Performance Software Administration or the **tqzip** program can be used to administer the workloads in your performance database. If you want to customize your workload definitions, you can perform the following administrative tasks:

- Create workload sets and definitions
- Modify workload definitions
- Delete workload sets and definitions
- Enable workload sets
- Disable workload sets

5.2. Concepts

The system activity monitor, TPM log probe, real-time TPM log probe, IOTRACE, MAPPER log, and system log probes use the workload definitions for creating parameters and assigning data values. Each probe has a unique set of workload characteristics. The language to specify the workload definition is the same for all workload probes. (See 9.4.4 for the system activity monitor workload characteristics, 9.7.5 for the RTPM and TPM workload characteristics, 9.8.3 for IOTRACE workload characteristics, 9.9.5 for MAPPER workload characteristics, and 9.10.3 for system log workload characteristics.)

The order that workloads are defined is very important. When a probe processes an input record, the workloads are evaluated in the order specified when they are defined. Once a probe determines that the data record belongs to a workload, the data is assigned to only that workload. The remaining workloads are not evaluated. If the record does not belong to any defined workload, it is assigned to a default workload named OTHER. The sum of all workloads, including OTHER, represents all the work performed on the system.

You define workloads using the workload definition language (see 5.2.2). To create workloads, you enter the definitions by using TeamQuest Performance Software Administration provided by TeamQuest Baseline or by updating a file unloaded by the **tqzip** program (see 5.4.1).

5.2.1. Workload Qualifiers

Each probe allows you to identify specific qualifiers for each workload set. Depending on the source data type, you can qualify the workload sets according to system name, environment name, or DCP name.

The system activity monitor, TPM, and system log probe workload sets can be qualified by system name. If no qualifying value is defined for the workload set, the workload set is assumed to apply to all systems whose data is analyzed by the system activity monitor or system log probes.

The MAPPER log probe workload sets can be qualified by system name and MAPPER environment name. If no system name is specified, the workload set is assumed to apply to all systems. If no MAPPER environment name is specified, the workload set definition is assumed to apply to all MAPPER environments whose data is analyzed by the MAPPER log probe.

5.2.2. Workload Definition Language

You use the workload definition language to specify the definitions of your workloads. This language is used when specifying workload definitions through TeamQuest View.

A workload definition is a Boolean combination of conditions. A definition is expressed as the combination of a characteristic name, an operator, and a value. This combination is referred to as a condition. A condition can be either a numerical condition or a string condition depending on the characteristic used in the definition. A numerical condition is used when the values for the characteristic must be numeric. A string condition is used when the values for the characteristic must be a string. Conditions can be combined using the Boolean operators OR, AND, and NOT to form complex definitions.

The following are examples of workload definitions:

```
PID = 111

Run_ID = "CMS"
```

5.2.2.1. Using Numerical Conditions

A numerical condition is expressed as a characteristic name, a numerical operator, and a numerical value set. An example of a numerical condition is

```
PID<=111
```

In this example, PID is the characteristic name, <= is the numerical operator, and 111 is the numerical value set.

When creating a numerical condition, the following rules apply:

- The characteristic name can be any of the characteristics associated with a specific probe listed as an integer or real number type. Examples of numerical characteristics are PID (system log probe) and Appl_Group (TPM log probe).
- Numerical operators can be =, !=, <, >, <=, and >=.
- A numerical value set is a single integer, a real number, or any numbers separated by commas and enclosed in braces ({ }).

Example 1

The following example will return values where PID is less than or equal to 123.

```
PID <=123
```

Example 2

The following example will return values where PID is equal to any of the values within the braces.

```
PID={1001,1012,1051,1022}
```

5.2.2.2. Using String Conditions

A string condition is expressed as a characteristic name, a string operator, and a string value set.

An example of a string condition is

```
User_ID="STF"
```

In this example, User_ID is the characteristic name, = is the string operator, and STF is the string value set. Run_ID, Site_ID, and Program are examples of string characteristic names associated with the system log probe.

When creating a string condition, the following rules apply:

- The characteristic name can be any of the characteristics associated with a specific probe listed as a string type.
- String operators can be = and !=.
- A string value set can be any of the following:
  - A single string with the characters enclosed in double quotation marks (“ ”) or in single quotation marks (‘ ’).
  - A single pattern consisting of a regular expression enclosed in slashes (/ /).
  - Several strings and patterns separated by commas (,) and enclosed in braces ({ }).

Example 1

The following example will match all occurrences where the string begins with TQ. The “.*” part of the string is significant where the period (.) is required with the asterisk (*) to wild-card match anything after TQ.

```
Run_ID =/TQ.*/
```

Example 2

The following example illustrates the matching of multiple strings separated by commas (,) and enclosed in braces ({ }).

```
Run_ID={"HALE",/TSTER[0-9]/,'SAA',"BDK"}
```

In the preceding example, any Run_ID satisfying any of the following strings or patterns will be returned.

`"HALE"`	matches only on the string HALE
`/TSTER[0-9]/`	matches on the string TSTER and a single numeric number 0 through 9
`'SAA'`	matches only on the string SAA
`"BDK"`	matches only on the string BDK

Example 3

A string condition containing a string enclosed in double quotation marks (“ ”) or single quotation marks (‘ ’) indicates that an exact match is required when the definition is evaluated. The following examples are interchangeable:

`Program="FURPUR"`	matches on the string FURPUR and only FURPUR
`Program='FURPUR'`	matches on the string FURPUR and only FURPUR

5.2.2.3. Using Regular Expressions in String Conditions

A string condition containing a string enclosed in slashes (/ /) indicates the use of regular expression matching. The string contained between the slashes can be made up of simple or compound regular expressions. For example,

`Txn_Code=/CSF/`	matches on the string CSF and only CSF
`Txn_Code=/CSF[0-2]/`	matches on the string CSF0, CSF1, and CSF2
`Txn_Code=/CSF.*/`	matches on any string beginning with CSF and followed by anything

Regular expressions are defined recursively with the following rules:

- A single character is a regular expression that matches the character itself. The following characters have special significance and do not match the character itself: a period (.), a backslash (\), brackets ([ ]), a hyphen (-), a caret (^), a plus sign (+), parentheses ( ), or an asterisk (*).
- A period (.) is a regular expression that matches any single character. For example, the period matches an A or B or any other character.

`Run_ID=/SLC./`	matches on 'SLC' and any single character following

- A backslash (\) followed by a single character is a regular expression that matches a single character. For example,

`User_ID=/SLC\./`	matches on the string ‘SLC.’. The backslash indicates that the period is treated as a character and is included in the match.

- A set of characters enclosed in brackets [...] is a regular expression that matches any one of the characters enclosed within the brackets. The regular expression [AB] matches A or B. For example,

`Txn_Code=/[CSF[ABC]/`	matches on CSFA or CSFB or CSFC
`Txn_Code=/CSF[XY]0/`	matches on CSFX0 or CSFY0

- A range of consecutive characters can be specified by using a hyphen (-) to separate the first and last characters desired. For example, [A-M] matches the characters A through M. The hyphen (-) loses this special meaning if it occurs first (after an initial caret [^], if any) or last in the string.

`Run_ID=/USER[A-E]/`	matches on USERA, USERB, USERC, USERD, or USERE

- If a caret (^) is the first character in the brackets, the regular expression matches any character except one of those found in the brackets. For example,

`Run_ID=/USER[^0-9] /`	matches any character following USER, but does not match 0 through 9

- *regexp1regexp2* is a regular expression that matches *regexp1* followed by *regexp2*. For example, [0-9][A-F] matches any digit 0 through 9 and any character A through F.

`Txn=/CSF[D-G][2-5]/`	matches on CSF followed by D through G and 2 through 5

- *regexp** is a regular expression that matches zero or more occurrences of the regular expression *regexp* that immediately precedes it. For example, A* matches a null string or A, AA, AAA, and so on.

`Run_ID=/RO*T/`	matches on anything beginning with R, then zero or any number of the character O, and ending with T. This matches RT, ROT, ROOT, ROOOT, and so on.

- *regexp*+ is a regular expression that matches one or more occurrences of the regular expression *regexp* that immediately precedes it. For example, A+ matches A, AA, AAA, and so on.

`Run_ID=/RO+T/`	matches on anything beginning with R, then any number of the character O, and ending with T. This matches ROT, ROOT, ROOOT, and so on.

- (*regexp*) is a regular expression that matches the regular expression *regexp*. This rule can be used to build complex regular expressions. For example, (AB)* matches a null string, AB, ABAB, ABABAB, and so on.

5.2.2.4. Using Boolean Expressions in Definitions

The Boolean operators OR, AND, and NOT can be used to combine numerical conditions and string conditions to form a definition. Parentheses can be used to override the default precedence. The following example shows the use of Boolean operators:

```
PID = 111 OR (Account = "10100" AND Run_ID=/TQ.*/)
```

In this example, the numerical condition PID = 111 must be true or both string conditions Account = “10100” and Run_ID = /TQ.*/ must be true.

The operators use the following order of precedence:

Operator	**Order of Precedence**
()	left to right
<,>,<=,>=,=,!=,==,<>	left to right
{}	left to right
!, NOT	right to left
&&, AND	left to right
||, OR	left to right

The operators = and == are equivalent.
The operators <> and != are equivalent.
The operators AND and && are equivalent.
The operators OR and || are equivalent.
The operators NOT and ! are equivalent.

5.2.2.5. Definition Example

The following example shows a definition using characteristics for run identifiers, account, and PID.

```
Run_ID = "VENCTL" OR
(Run_ID = /VEN.*/ AND (Account = /102[0-9]9/ AND Account != "10229"))
OR
PID = 111 OR (PID >= 130 AND PID <= 135)
```

This workload contains information for the following:

- Run VENCTL
- Runs that begin with VEN and have account numbers 10209, 10219, and 10239 through 10299
- PID 111 and 130 through 135

5.3. Maintaining Workloads with TeamQuest Performance Software Administration

To perform any administration tasks with TeamQuest Performance Software Administration, you must have a session established to the host software (see 4.2). You access the dialog boxes for performing administration tasks by selecting the Administration menu.

Note: *When maintenance tasks are being performed, the database is locked. Only the user performing the tasks can access the database for modifying or updating workloads. Other users can view workload information in read-only mode when the database is locked.*

TeamQuest Performance Software Administration allows you to perform the following tasks:

- Performing Workload Administration Tasks (see 5.3.1)
- Creating a New Workload Set Definition (see 5.3.2)
- Displaying or Modifying a Workload Set Definition (see 5.3.3)
- Deleting a Workload Set (see 5.3.4)
- Creating a New Workload (see 5.3.5)
- Displaying or Modifying a Workload Definition (see 5.3.6)
- Importing Workload Sets and Workload Definitions (see 5.3.7)
- Exporting Workload Sets and Workload Definitions (see 5.3.8)
- Deleting a Workload (see 5.3.9)

5.3.1. Performing Workload Administration Tasks

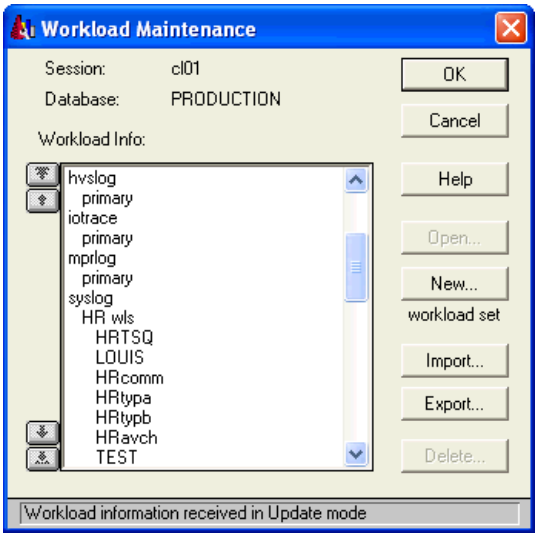
To begin any workload maintenance task, select Administration > Workloads.

If more than one session is open, the Administration Session Selection dialog box is displayed. This dialog box contains a list of open sessions for which you can perform workload maintenance and session information for the session highlighted in the list box. Select the desired session and click OK.

If more than one database is available for the session, the Administration Database Selection dialog box is displayed. Select the database for which you want to perform workload maintenance and click OK.

The Workload Maintenance dialog box is displayed.

Note: *As the workload information is loading, the status bar at the bottom of the dialog box indicates the percentage of information loaded. When the percentage reaches 100, the message disappears and you are ready to perform workload maintenance tasks.*



All workload maintenance tasks can be performed through this dialog box. The Workload Info list box contains the list of source data types, workload sets, and workloads defined for the database and session listed at the top of the dialog box.

The first item in the Workload Info list box is a source data type. Source data types are defined when a probe is registered with the database at installation time. They automatically appear in the Workload Info list box. Source data types cannot be modified and, therefore, the Open and Delete buttons do not apply to the source data types.

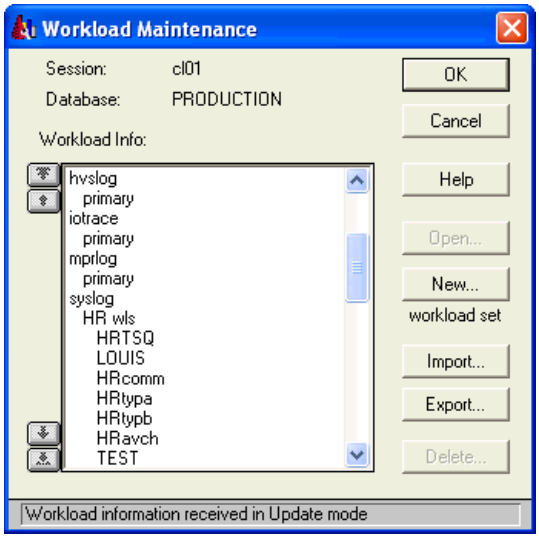
The items listed under the source data type are the workload sets and the workloads. Workload sets are listed at the first indentation under the source data type, and workloads are indented under the workload set to which they belong.

The status bar at the bottom of this dialog box displays information about the selected item in the Workload Info list box. It tells you the type of item—source data type, workload set, or workload—that is selected from the list.

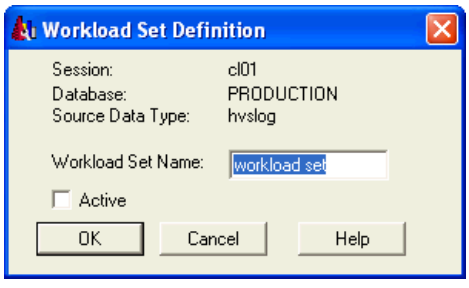
You can use the arrows to the left of the Workload Info list box to rearrange the workloads within the workload set for which they are defined. Select the workload and click the up or down arrow to move the workload up or down the list. See 5.2 for a description of how the order of the workloads affects the processing of data.

5.3.2. Creating a New Workload Set Definition

To create a new workload set definition, select the source data type from the Workload Info list box in the Workload Maintenance dialog box. Notice that the term *workload set* appears under the New button indicating that you are going to insert a workload set definition into the database.



When you click the New button, the Workload Set Definition dialog box is displayed.



Workload Set Definition Dialog Box

The Workload Set Definition dialog box contains information about the session, database, and source data type for which you are creating a workload set definition. It also contains the following fields:

Workload Set Name
: supplies a name for the workload set. A default name is supplied from TeamQuest Baseline. You can use this name or supply your own name. This field is limited to 51 characters in length.

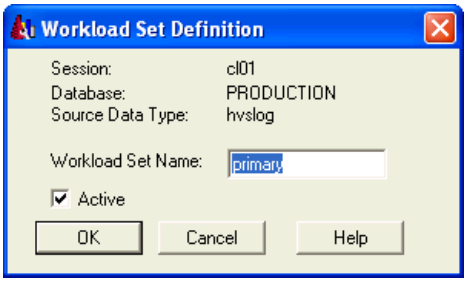
Active
: specifies the status of the workload set. A check in the active box indicates that the workload set is active and is available for receiving data from the data collection probes. If this box is not checked, the workload set is inactive and does not receive data.

When your definition is complete, click OK. The new workload set is inserted into the Workload Info list box in the Workloads Maintenance dialog box. The workload set appears after the item selected in the Workload Info list box.

To define the workload set to the host software, you must click OK in the Workload Maintenance dialog box. Your definition is not sent to the host software until you click OK.

5.3.3. Displaying or Modifying a Workload Set Definition

To display or modify a workload set definition, select the workload set from the Workload Info list box in the Workload Maintenance dialog box (see 5.3.1) and click Open. You can also double-click on the workload set name in the Workload Info list box. The Workload Set Definition dialog box is displayed.



You can use this dialog box to modify the workload set name, active status, and qualifiers. For a description of the fields in this dialog box, see 5.3.2.

When you click OK, your changes are accepted and you return to the Workload Maintenance dialog box (see 5.3.1). Your changes are not sent to the host software until you click OK in the Workload Maintenance dialog box.

5.3.4. Deleting a Workload Set

To delete a workload set from a database, you select the workload set name from the Workload Info list box in the Workload Maintenance dialog box (see 5.3.1). When you click the Delete button, an alert box verifies your request to delete the workload set.

When you click Yes in this alert box, the workload set name is removed from the Workload Info list box in the Workload Maintenance dialog box. This causes the selected workload set and all associated workloads to be deleted. To delete the workload set definition from the database, you must click OK in the Workload Maintenance dialog box.

You can cancel the deletion of a workload set either by clicking the No button in the alert box or by clicking Cancel in the Workload Maintenance dialog box (see 5.3.1).

Note: *When you delete a workload set, the database parameters associated with that workload set are also deleted. By deleting parameters from the database, all of the data values for these parameters are cleared in each of the aggregation sets in the database. For this reason, you must ensure that all of the database aggregation set files are loaded onto the system and are accessible whenever you are deleting workload sets. All space previously used by deleted parameters is automatically reused as new parameters are created.*

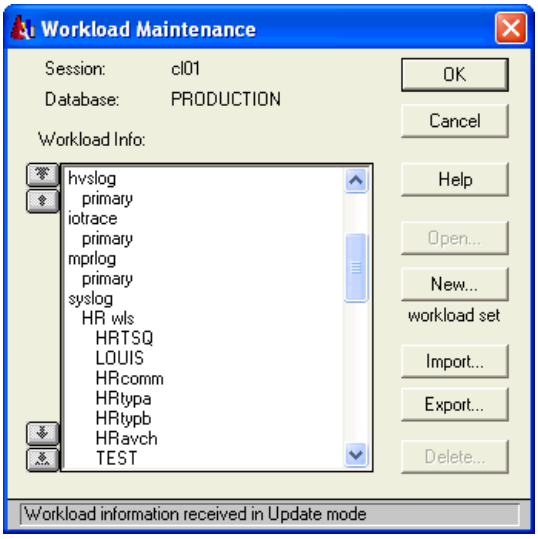
Deletion of workload sets should be a controlled operation and is best performed after you have terminated all probes executing on your system. Be careful not to delete any parameters that are being used by a probe that is currently executing on your system.

5.3.5. Creating a New Workload

To create a new workload definition,

1. Select a workload set or a workload from the Workload Info list box in the Workload Maintenance dialog box.

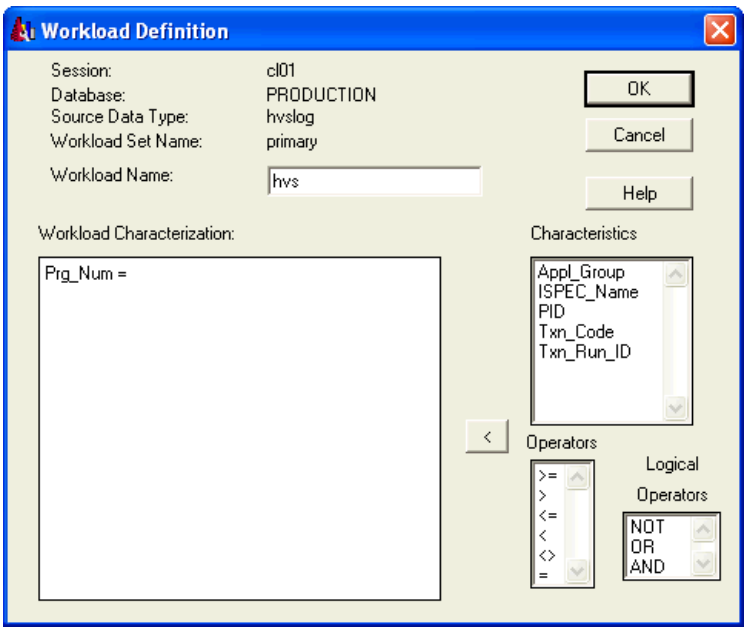
   The workload you define will be inserted after the selected item and belong to the selected workload set. Notice that the term *workload set* appears under the New button indicating that you are going to create a workload definition in the database.



The order that workloads are defined is very important. When a probe processes an input record, the workloads are evaluated in the order specified when they are defined. Once a probe determines that the data record belongs to a workload, the data is assigned to only that workload. The remaining workloads are not evaluated. If the record does not belong to any defined workload, it is assigned to a default workload named OTHER. The sum of all workloads, including OTHER, represents all the work performed on the system.

Note: *You can use the order of your workload definitions to streamline the evaluation process. If workloads that catch the most records are placed toward the top of the workload definition list, fewer evaluation steps are required and the evaluation process is completed more quickly.*

When you click New, the Workload Definition dialog box is displayed.



2. Select the characteristic and click the arrow button.

   The characteristic then appears in the Workload Characterization edit box.

3. Select the operator from the Operators list box and click the arrow button.

   The operator appears in the Workload Characterization edit box.

4. Enter the numerical condition or string condition to complete the definition.

   If you want multiple definitions, use the Logical Operators list box to select an operator and enter another workload definition statement. You can also double-click the characteristics, operators, and logical operators to move them between the list boxes and the Workload Characterization edit box.

Workload Definition Dialog Box

The Workload Definition dialog box contains information about the session, database, source data type, and workload set name for which you are creating a workload definition. It contains the following fields:

Workload Name
: supply a name for the workload. The workload name is limited to 51 characters in length.

Workload Characterization
: enter the workload definition. The workload definition must be expressed using the workload definition language. You can type in the definition directly into the edit box or you can use the Characteristics, Operators, and Logical Operators list boxes. See 5.2.2 for more information on workload characterization and the workload definition language.

Note: *The characteristics in the Characteristics list box are case-sensitive. These characteristics must appear in the Workload Characterization edit box exactly as they appear in the Characteristics list box.*

To complete a workload, you must enter a workload characterization. If the field does not contain a value, an error message is displayed. A syntax check will be done on the workload characterization for verification.

Characteristics, Operators, and Logical Operators
: These list boxes aid you in creating the workload definition. You can select an item from one of these boxes and then click the arrow button to transfer the item to the Workload Characterization edit box. You can also double-click the item to transfer it.

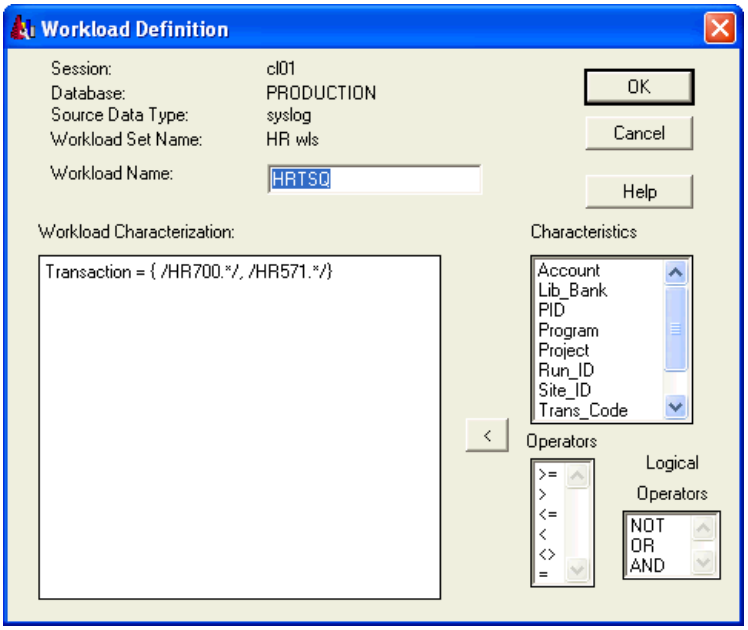
  Characteristics list box
  : contains a list of the characteristics defined for the source data type.

  Operators list box
  : contains a list of the operators allowed for the workload definition language.

  Logical Operators list box
  : contains a list of the logical operators allowed for the workload definition language.

5.3.6. Displaying or Modifying a Workload Definition

To display or modify a workload definition, select the workload from the Workload Info list box in the Workload Maintenance dialog box (see 5.3.1) and click Open. You can also double-click the workload name in the Workload Info list box. The Workload Definition dialog box is displayed.



You can use this dialog box to modify the workload name and workload characterization. For a description of the fields in this dialog box, see 5.3.5.

When you click OK, your changes are accepted and you return to the Workload Maintenance dialog box (see 5.3.1). Your changes are not sent to the host software until you click OK in the Workload Maintenance dialog box.

5.3.7. Importing Workload Sets and Workload Definitions

To import all workload set and workload definitions, click the Import button on the Workload Maintenance dialog box (see 5.3.1). When the Open File dialog box is displayed, select the file that contains the workload set and workload definitions that you want to import.

When the workload set and workload definitions are imported, all existing workload sets and workload definitions are updated to match the imported definition. If you deleted a workload definition from the exported file and then imported the remaining workload definitions, the deleted workload definitions will be re-ordered to the last position. Use the Delete button to remove the workload definitions. New workload sets and workload definitions are inserted into the Workload Maintenance dialog box.

If a syntax error occurs, the workload set and workload definitions are not inserted in the Workload Maintenance dialog box.

When importing new workloads with existing workload sets, it is important to check the order of the workloads within the existing workload sets.

After an import function, your updates can be deleted by clicking the Cancel button on the Workload Maintenance dialog box. To accept the updates, click OK and all updates are written to the session database.

See 5.4.1 for an example of a workload definition file.

5.3.8. Exporting Workload Sets and Workload Definitions

To export all workload sets and workload definitions, click the Export button on the Workload Maintenance dialog box (see 5.3.1). When the Save File dialog box is displayed, select or enter a file that will contain a workload definition for all workload sets and all workloads listed in the Workload Maintenance dialog box.

See 5.4.1 for an example of a workload definition file.

5.3.9. Deleting a Workload

To delete a workload from a database, select the workload name from the Workload Info list box in the Workload Maintenance dialog box (see 5.3.1). When you click the Delete button, an alert box verifies your request to delete the workload.

When you click Yes in this alert box, the workload name is removed from the Workload Info list box in the Workload Maintenance dialog box. To delete the workload definition from the database, you must click OK in the Workload Maintenance dialog box.

You can cancel the deletion of a workload either by clicking the No button in the alert box or by clicking Cancel in the Workload Maintenance dialog box (see 5.3.1).

Note: *By deleting a workload definition from the database, neither any database parameters are deleted nor are any data values cleared. This is because deleting individual workload parameters and data values would cause the data previously stored for that workload set to be incomplete.*

5.4. Maintaining Workloads with tqzip

The **tqzip** program allows you to perform the following additional tasks associated with maintaining the workload capability:

- Unload workload definitions contained in the database to a file (see 5.4.1). You can use this file to view the definitions and update the definitions by using a text editor. You can then reload the definitions.
- Load workload definitions from a specified file into the database (see 5.4.2).

5.4.1. Unloading Workload Definitions

The **tqzip** program allows you to export all the workload definitions currently contained in the performance database into a specified file. You can use this file to view the workload definitions and update the workload definitions by using a text editor. You can then reload the workload definitions by using the -l and -w options on the **tqzip** processor call (see 5.4.2).

To unload workload definitions with **tqzip**, use the following processor call format:

```
@tqzip [-d,databasename,] -u,-w,workload-file
```

where:

-d specifies the database that **tqzip** uses. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-u specifies that an unload operation is to be performed by **tqzip**. The type of unload operation is "workloads" as indicated by the -w option.

-w unloads the workload definitions from the database to the file specified by the *workload-file* parameter.

Example

The following shows an example of unloading workload definitions to the file wfile.

```
>@tqzip -u,-w,wfile
>TQZIP 7R4G (011312 1104:55) Fri Jan 13 15:13:19 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>
>Using database TQDATA:PRODUCTION
>
>Workload definitions unloaded.
>
>End TQZIP
```

Workload Definition File Format

The workload set and workload definitions appear in the following format in the output file created by **tqzip**. If you choose to edit this file or update the definitions, you should maintain this file format.

```
BEGIN WORKLOADS unloaded on Fri Jan 14 10:54:13 2012                A
'syslog' 'primary' Active 1                                         B
'system' 3                                                          C
'venus'                                                             D
'saturn'
'mars'
'syslog' 'secondary' Active 1                                       E
'system' 1
'saturn'
                                                                    F
'syslog' 'primary' 'sysjob' 001                                     G
"                                                                   H
'Run_ID=/RUNNE./ || Run_ID=/MSCP.*/ || Run_ID=/CMR.*/ ||'           I
'Run_ID=/PCCTI.*/ || Run_ID=/DDP.*/ || Run_ID=/PXFR.*/'

'syslog' 'primary' 'cmsmcb' 002                                     J
"
'Run_ID=/CMS.*/ || Run_ID=/MCB.*/'

'syslog' 'primary' 'accounting' 003
"
'Run_ID=/ACC.*/ || Account=/1010.*/'

'syslog' 'primary' 'venus' 004
"
'Run_ID=/VEN.*/ || Account=/2190.*/ || Account=/1103.*/'

'syslog' 'secondary' 'sysjob' 001
"
'Run_ID=/RUNNE./ || Run_ID=/MSCP.*/ || Run_ID=/CMR.*/ ||'
'Run_ID=/PCCTI.*/ || Run_ID=/DDP.*/ || Run_ID=/PXFR.*/'

'syslog' 'secondary' 'accounting' 002
"
'Run_ID=/ACC.*/ || Account=/3333.*/'
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A The first line of the workload definition file must be the header line BEGIN WORKLOADS. A blank line must follow the header line.

B The first section following the header line contains a list of the workload set definitions. Each workload set definition consists of the source data type, the workload set name, the Active/Inactive flag, and the number of workload set qualifiers for the workload set. In this example, syslog is the source data type; primary is the workload set name; the active flag is set; and the number of workload set qualifiers is 1.

C The second line of the workload set definition contains the name of the first workload set qualifier and the number of qualifier values for the workload set qualifier. The qualifier is System and there are three qualifier values.

D Each qualifier value for the workload set is then listed on a line by itself. Venus, saturn, and mars are the qualifier values for this workload set.

E The second workload set definition begins on this line. syslog is the source data type; secondary is the workload set name; the active flag is set; and the number of qualifiers is 1. The second line of the workload set definition shows that System is the qualifier and it has one qualifier value.

F A blank line separates the list of workloads sets from the list of workloads. Each workload consists of at least three lines, with a blank line separating one workload from the next.

G The first line of each workload contains the source data type, the workload set name, the workload name, and the workload sequence number. syslog is the source data type; primary is the workload set name; sysjob is the workload; and 001 indicates that this is the first workload in the primary workload set.

H The second line of the workload contains descriptive text describing the workload. This text must be enclosed in single quotation marks (' '). For this example, the text is not present.

I The third line of the workload begins the actual workload definition specified in the workload definition language (see 5.2.2). The workload definition can continue on subsequent lines and is terminated by a blank line. Each line of the workload definition language must be enclosed in single quotation marks (' '). Notice that single quotation marks are also used within the braces to enclose text strings.

J This line begins the cmsmcb workload, which is the second workload of the primary workload set. Following the cmsmcb workload are the rest of the workloads in the primary workload set. Then the workloads in the secondary workload set are listed.

The following general rules apply for the workload definition file:

- Each text field of the workload set and workload definitions must be enclosed in single quotation marks (' '). However, text strings specified within the workload definition expression itself can be enclosed in single quotation marks (' ') or double quotation marks (" ").
- Do not use quotation marks around the Active/Inactive flag (line B), workload set qualifier count (line B), or workload sequence number (line G).
- Workload sequence numbers must be sequential within a workload set and do not require leading zeros as seen in the preceding example (see line G).
- Blank lines are significant and are used to separate the header from the first workload set definition, the workload sets from the first workload, and subsequent workloads from each other.

5.4.2. Loading Workload Definitions

You can reload the performance database with a new or updated set of workload definitions by using the -l and -w options on the **tqzip** processor call. You should use the file as created by the -u and -w options (see 5.4.1) as input for loading workload definitions. If you update the workload definition file, you should ensure that the format of the file remains consistent with what was output by the -u and -w options.

Note: *The **tqzip** program deletes any existing workload definitions from your database before loading the new definitions. Use the -e option on **tqzip** if you do not want existing definitions deleted.*

To load workload definitions with **tqzip**, use the following processor call format:

```
@tqzip [-d,databasename,] -l,-w,workload-file [-e]
```

where:

-d specifies the database that **tqzip** uses. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-l specifies that a load operation is to be performed by **tqzip**. The type of load operation is “workloads” as indicated by the -w option.

-w loads the workload definitions into the database from the file specified by the *workload-file* parameter. The *workload-file* is the name of a workload definition file created by the **tqzip** -u and -w options (see 5.4.1).

-e loads the new workload definitions from the workload definition file without first deleting the existing definitions in the database. If you do not specify the -e option, all existing definitions are deleted from your database before the new ones are loaded.

Example

The following shows an example of loading workload definitions from the file wfile.

```
>@tqzip -l,-w,wfile
>TQZIP 7R4G (011312 1104:55) Fri Jan 13 14:01:11 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>
>Using database TQDATA:PRODUCTION
>Workload Set          'workloadset1' has been inserted/modified
>Workload Set          'workloadset2' has been inserted/modified
>Workload              'workload1' has been inserted/modified
>Workload              'workload2' has been inserted/modified
>Workload              'workload1' has been inserted/modified
>Workload              'workload2' has been inserted/modified
>Workload load complete.
>
>Workload definitions loaded.
>
>End TQZIP
```

Section 6
Customizing Aggregation Sets

6.1. Introduction

TeamQuest Performance Software Administration or the **tqasa** program can be used to administer the aggregation sets in your performance database. Default aggregation sets are created during installation. If you are content with the default aggregation sets, no additional administration is required. However, if you want to customize your aggregation sets, you can perform the following administrative tasks:

- Create aggregation sets
- Remove aggregation sets
- List aggregation sets
- Unload aggregation set definitions
- Load aggregation set definitions
- Convert expanded aggregation sets
- Perform data aggregation

6.2. Concepts

The performance database consists of data records saved in various aggregation sets. With TeamQuest Baseline, you can define additional aggregation sets for your database. There are some basic concepts essential to the understanding of TeamQuest Baseline aggregation sets. Before defining additional aggregation sets, you will want to know about the aggregation set structure, the moving windows concept for data storage, how aggregation sets are consolidated, feeding aggregation sets with data from other aggregation, and compact/expanded aggregation sets.

6.2.1. Aggregation Set Structure

Granularity, date grouping, retention period, and start and end time are concepts essential to the understanding of TeamQuest Baseline aggregation sets. To define an aggregation set, you specify the following information:

- The granularity of the data records within the aggregation set

  Specify the time interval represented by each data point maintained in the database for the aggregation set. This interval is called the granularity. For example, you can specify that TeamQuest Baseline collect and store data samples at 30-minute intervals, 1-hour intervals, or perhaps, 8-hour intervals. Each data value in an aggregation set represents the value for a parameter over the specified time interval.

- A type of date consolidation

  Specify the type of date consolidation represented by each point maintained in the database for the aggregation set. TeamQuest Baseline supports date consolidation types of day, week, and month. For example, you can have an aggregation set that stores data records with 30-minute granularity throughout each day. A single data point exists for each 30-minute interval for each day the data was collected. Another aggregation set can be defined that still has a granularity of 30 minutes, but where each data point represents a 1-week or 1-month average for each 30-minute interval.

  For the week and month date consolidation types, you can choose days of the week to exclude from the aggregation set. For the week date consolidation type, you can also choose the day of the week that is considered to be the first day of the week.

- The retention period for the data

  Specify the amount of time to maintain the data in the database for the aggregation set. For example, you could retain your data for one week, two months, or one year. The retention period determines the amount of data in your database. The longer you retain your data, the larger your database will become.

- The start and end time of data sampling

  Specify the time range for sampling data. For example, you can specify that data is sampled from 08:00 to 15:59 or that it is sampled for a complete 24-hour interval from 00:00 to 23:59. From this time interval, TeamQuest Baseline calculates the number of time groupings within each type. If you collect data for 24 hours each day and the data has a granularity of 30 minutes, you will have 48 time groupings for each day.

Figure 6–1 shows the logical layout of how the data records for a parameter are grouped in an aggregation set. The aggregation set in the figure is defined to collect and store data samples at 30-minute intervals daily and retain that data in the database for a period of 60 days. Data is collected for a 24-hour period. So for this example, the granularity is 30 minutes; the type of date consolidation is day; the retention period is 60 days; the start time is 00:00:00; the end time is 23:59:59. Only one parameter, Parameter N, is shown in this figure. However, the aggregation set would contain separate groupings for each parameter collected and stored.

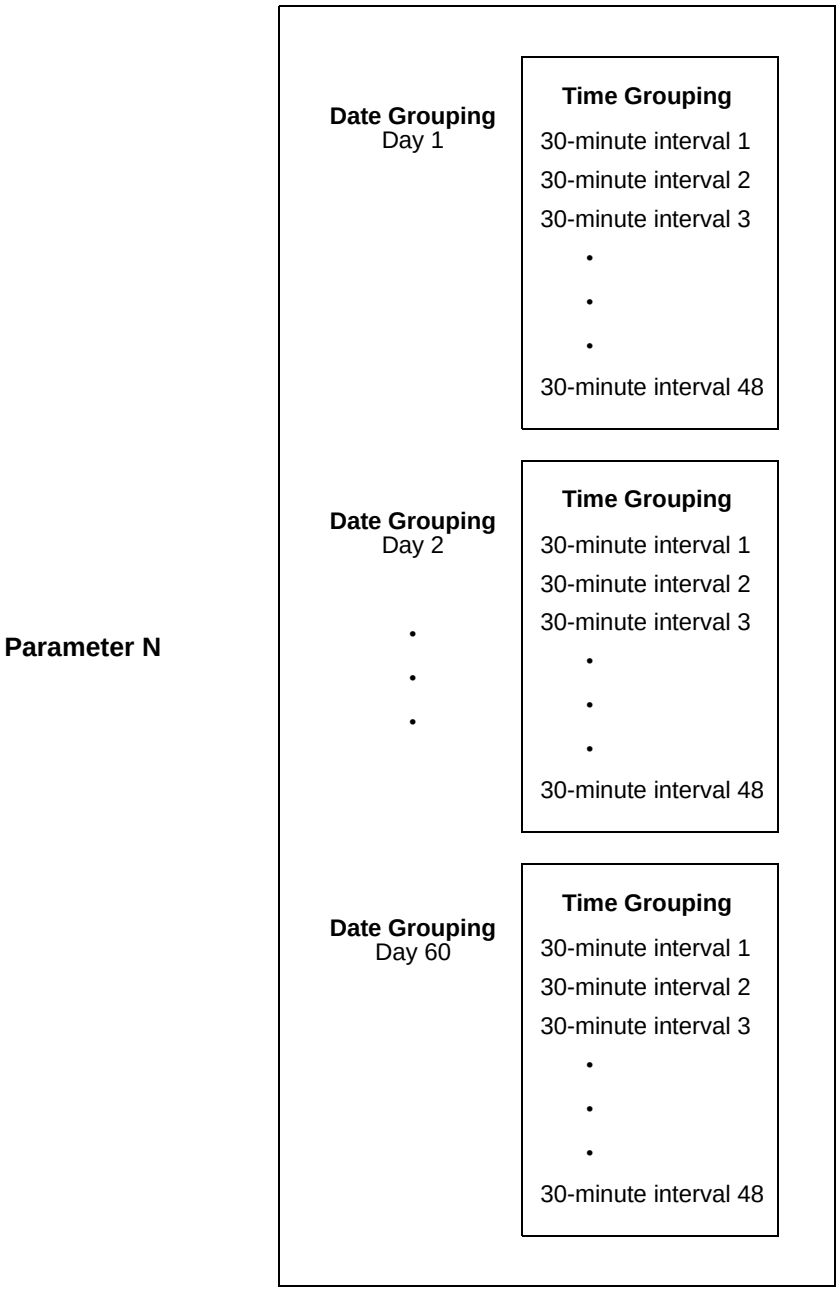


Figure 6–1. Parameter Grouping for an Aggregation Set

6.2.2. Moving Window Concept for Data Storage

The performance database uses a moving window concept for storing data. It begins storing data in the aggregation set at the start date of the database, usually the current date. It stores the data records having the required date and timestamp within the date grouping. TeamQuest Baseline fills in data for the specified number of date groupings. When it has filled in the last date grouping, it removes the data in the first date grouping. When you specify the retention period for an aggregation set, you are actually specifying the size of the moving window for the aggregation set.

In the example in Figure 6–1, TeamQuest Baseline records data at 30-minute intervals for date grouping 1. Each day, it fills in another date grouping. When it has filled in date grouping 61, TeamQuest Baseline removes all day 1 data; when it has filled in date grouping 62, it removes all data for day 2; and so on.

6.2.3. Aggregation Set Consolidation

Note: *The data aggregation capability is not available with TeamQuest Online.*

The TeamQuest Baseline capability to support multiple aggregation sets in a database provides a way for consolidating data stored at a fine granularity (for example, a day date consolidation type at 30-minute intervals) to a coarser granularity (a week date consolidation type at 1-hour intervals). The aggregation sets can reduce the amount of data that is retained in the database at the expense of losing some of the detail of the data as it was originally created. When data is consolidated from one aggregation set into another aggregation set, the consolidation can occur in two dimensions:

- Time consolidation

  You can consolidate your data using time consolidation. For example, you might have data that is collected at 15-minute intervals and wish to consolidate this data to a granularity of 60 minutes. The original aggregation set has a granularity of 15 minutes. You can create a second aggregation set that has a granularity of 60 minutes. The data from the 15-minute aggregation set is used as the source for the 60-minute aggregation set. Each of the four 15-minute points from the 15-minute aggregation set are consolidated into one point of data for the 60-minute aggregation set (the four data points for 10:00 to 10:59 in the original aggregation set are consolidated into one data point for 10:00 to 10:59 in the 60-minute aggregation set).

  The 15-minute aggregation set is the source of the data for the 60-minute aggregation set. The 60-minute aggregation set is said to be fed by the 15-minute aggregation set. When you define an aggregation set, you can optionally specify the aggregation set that is the source of the data (that is, the aggregation set that feeds the newly created aggregation set).

- Date consolidation

  The second dimension of consolidation is date consolidation. For example, you might have data that is collected at 30-minute intervals throughout each day. A single data point exists for each 30-minute interval. You can create a second aggregation set that still has a granularity of 30 minutes, but that has a group type of a week. In this second aggregation, a single data point represents the 30-minute interval for the week. In the first aggregation set (the day consolidation type), the data point for 10:00 to 10:29 represents the value for that interval for the day. In the second aggregation set (the week consolidation type), the data point for 10:00 to 10:29 represents the value of the seven data points for each day of the week.

  When you are consolidating data by days, you might want to ignore certain days of the week so that you are accurately representing the normal work days of the system users. For this reason, TeamQuest Baseline allows you to exclude certain days of the week when you consolidate data from a day consolidation type into a week or month consolidation type. When you create such an aggregation set, you must specify the days to ignore in the aggregation set definition. TeamQuest Baseline normally assumes that all days will be included unless you specify the days to exclude.

  TeamQuest Baseline allows you to specify the day of the week that should be considered the first day of the week. TeamQuest Baseline normally assumes that Sunday is the first day of the week unless you specify another day as the first day of the week.

You use TeamQuest Performance Software Administration to create the aggregation sets for data consolidation. To consolidate data from one aggregation set to another, you must use the **tqagg** program (see 6.4.7).

Figure 6–2 illustrates the concept of consolidating data from one aggregation set to another. In this example, the performance database has two aggregation sets defined, each containing information collected 24 hours each day. The first aggregation set is the same as shown in Figure 6–1. It contains data records saved for every parameter at 30-minute intervals: a total of 48 data records for each parameter of each day. The second aggregation set saves the data records at 1-hour intervals weekly for 52 weeks. In this aggregation set, TeamQuest Baseline saves a total of 24 data records for each parameter summarized by week. At the end of each week, TeamQuest Baseline automatically rolls the data from the first aggregation set into the second aggregation set consolidating seven daily groups (each with 48 time groupings) into one weekly group (with 24 time groupings).

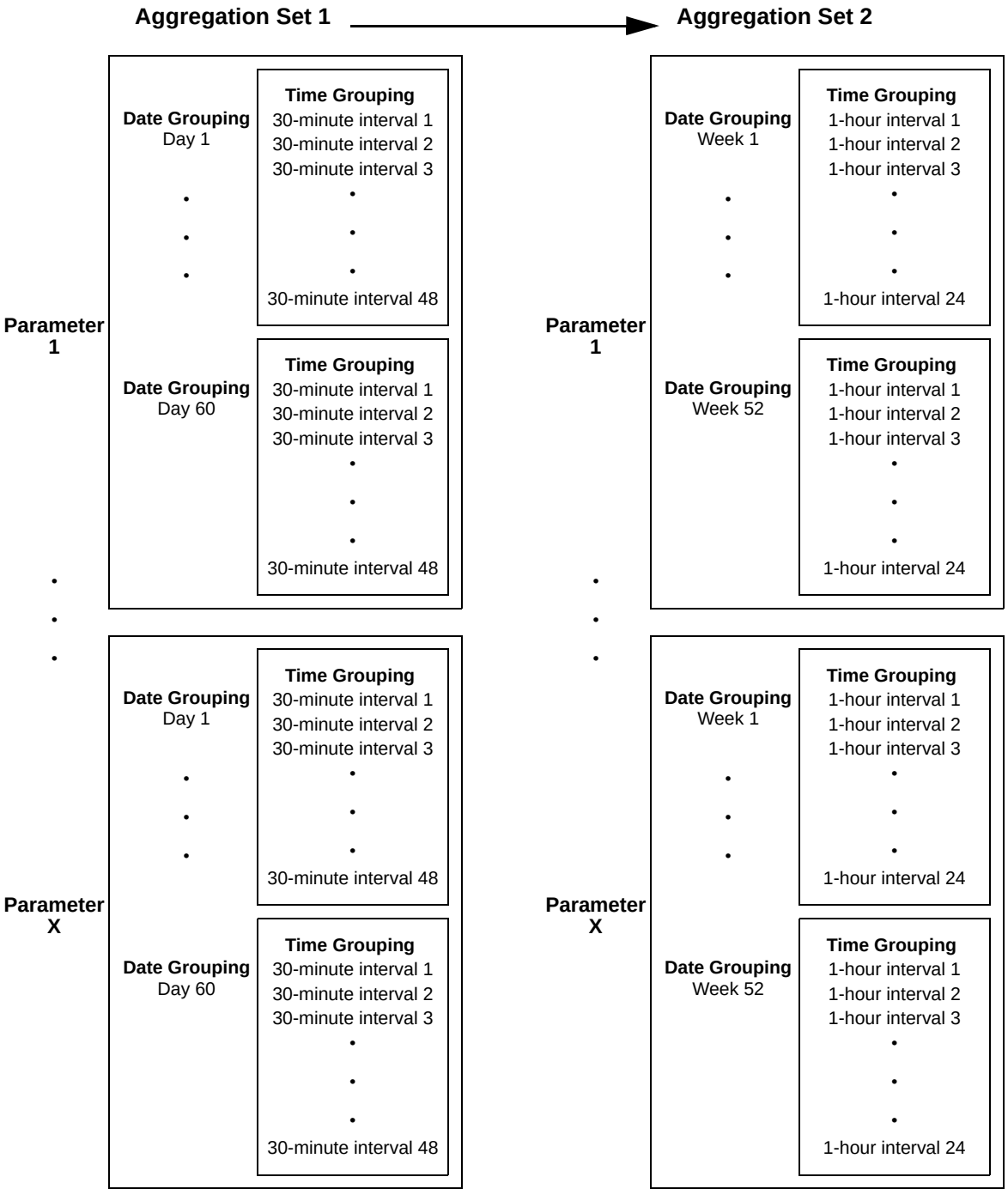


Figure 6–2. Data Consolidation from Aggregation Set 1 to Aggregation Set 2

6.2.4. Feeding Aggregation Sets

When an aggregation set receives its data from another aggregation set, the second aggregation set is said to be fed by the first aggregation set. For example, if data from Aggregation-Set 1 is consolidated into Aggregation-Set 2, Aggregation-Set 1 is the source of data for Aggregation-Set 2. Aggregation-Set 2 is fed by Aggregation-Set 1. There are a number of rules to consider when determining if an aggregation set can or should be fed by another aggregation set.

- The granularity of the fed aggregation set must be greater than or equal to the granularity of the source aggregation set. A 15-minute aggregation set can feed a 60-minute aggregation set, but a 60-minute aggregation set cannot feed a 15-minute aggregation set.
- The granularity of the fed aggregation set must be an even multiple of the granularity of the source aggregation set. For example, a 5-minute aggregation set can feed a 30-minute aggregation, 30 minutes is an even multiple of 5 minutes.
- The date consolidation type of the fed aggregation set must be larger than or the same as the feeding aggregation set. For example, a week consolidation type can feed another week consolidation type, but it cannot feed a day consolidation type.
- A week consolidation type cannot feed a month consolidation type, as most months do not have an even number of weeks in them.
- The days to exclude must match between the fed aggregation set and the source aggregation set.
- If a week consolidation type feeds another week consolidation type, the start day of the week must be the same for both aggregation sets.
- If the source aggregation set does not retain data for a period that is long enough to encompass a single point in the fed aggregation set, the fed aggregation set data will not accurately reflect the specified interval. If, for example, you feed a 1-month aggregation set from a day aggregation set that contains only 14 days worth of data, each point in the month aggregation set will reflect an average for the last 14 days of the month, not the average for the entire month.

6.2.5. Compact and Expanded Aggregation Set Format

Aggregation set data is stored in disk files on the host system. The aggregation set files can be in either compact or expanded format. By default, all aggregation sets created using TeamQuest Performance Software Administration or the **tqasa** program are in compact format.

The advantage of compact format is that significantly less mass storage is required to store the aggregation set data. Data points are stored as single-precision floating-point values in compact aggregation sets. Compact aggregation sets provide accurate data values while using minimal disk space to store data.

With compact aggregation sets, no standard deviation information is saved with each data point. You can only report the standard deviation when you are using time and/or date consolidation of data points at report time. When using the **tqagg** program to aggregate data from a compact aggregation set into an expanded aggregation set of a coarser granularity, standard deviation information is inserted into the expanded aggregation set as each point is written.

An expanded aggregation set provides more complete reporting of standard deviations than with compact aggregation sets. This is because information is saved with each data point allowing the standard deviation to be calculated at report time. Also, there is increased accuracy for very large or very small data point values since double-precision floating point is used to store the values in an expanded aggregation set.

Note: *Before level 6R2, all aggregation sets created using TeamQuest View or the* ***tqasa*** *program were in expanded format. You can convert existing expanded aggregation sets into compact aggregation sets by using the* ***tqasa*** *program (not available with TeamQuest Online). This conversion will result in a 40% to 60% reduction in the disk space required by the aggregation set. See 6.4.6 for information on how to convert expanded aggregation sets into compact aggregation sets.*

6.3. Maintaining Aggregation Sets with TeamQuest Performance Software Administration

To perform any administration tasks using TeamQuest Performance Software Administration, you must have a session established to the host software (see 4.2). You access the dialog boxes for performing administration tasks by selecting the Administration menu.

Note: *When aggregation set maintenance tasks are being performed, the database is locked. Only the user performing the tasks can access the database for creating new aggregation sets. Other users can view aggregation set information in read-only mode when the database is locked.*

TeamQuest Performance Software Administration allows you to perform the following tasks:

- Performing Aggregation Set Administration Tasks (see 6.3.1)
- Creating an Aggregation Set (see 6.3.2)
- Displaying or Modifying Aggregation Set Definitions (see 6.3.3)
- Deleting an Aggregation Set (see 6.3.4)

Note: *It is recommended that you use TeamQuest Performance Software Administration to maintain your aggregation sets.*

After you create, remove, or change an aggregation set, you must customize your data collection to take advantage of any aggregation changes (see 6.4.7).

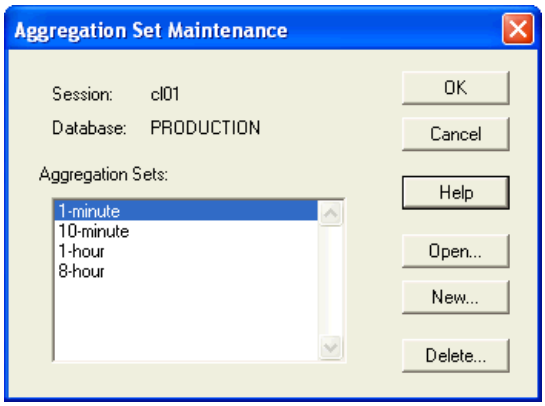
6.3.1. Performing Aggregation Set Administration Tasks

To begin any aggregation set maintenance task, select Administration > Aggregations.

If more than one session is open, the Administration Session Selection dialog box is displayed. This dialog box contains a list of open sessions for which you can perform aggregation set maintenance and session information for the session highlighted in the list box. Select the desired session and click OK.

If more than one database is available for the session, the Administration Database Selection dialog box is displayed. Select the database for which you want to perform aggregation set maintenance and click OK.

The Aggregation Set Maintenance dialog box is displayed.



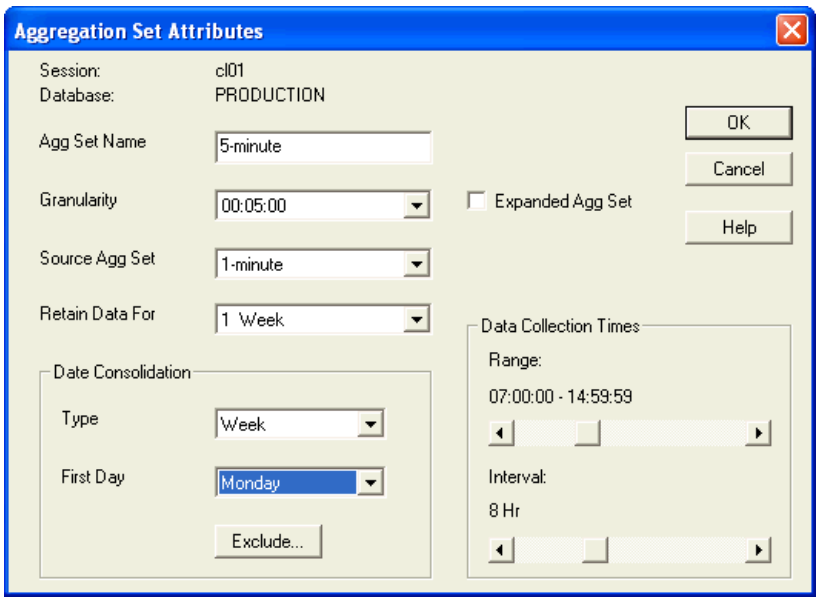
All aggregation set maintenance tasks can be performed through this dialog box. The session name and the database for which aggregation set maintenance is being performed are listed at the top of the dialog box. The Aggregation Sets list box contains the list of aggregation sets defined for the database. This dialog box is displayed with an aggregation set automatically selected. All operations are performed relative to the selected aggregation set. The Open button opens the selected aggregation set. The New button inserts an aggregation set after the selected item. The Delete button deletes the selected aggregation set. Select an aggregation set from the list box before performing any of the operations with the Open, New, or Delete buttons.

When you click OK, your changes are made to the aggregation set definitions, including any changes made to aggregation set attributes accessed through the Open or New buttons (see 6.3.2 and 6.3.3).

When you click Cancel, any changes you have made are ignored.

6.3.2. Creating an Aggregation Set

To create an aggregation set definition, click the New button in the Aggregation Set Maintenance dialog box (see Performing Aggregation Set Administration Tasks6.3.1). The Aggregation Set Attributes dialog box is displayed. You define a new aggregation using this dialog box.



Aggregation Set Attributes Dialog Box

The Aggregation Set Attributes dialog box contains the following fields:

Agg Set Name
: specifies the name of the aggregation set. A default name is supplied by TeamQuest Baseline. You can use this name or supply your own name. This field is limited to 23 characters in length.

Granularity
: contains a list of the available options for the granularity of the aggregation set. Granularity is the sample interval across which each data sample is taken. Select the granularity for the new aggregation set from this list box.

Source Agg Set
: contains a list of aggregation sets already defined in the database that can be used as the source of data for the aggregation set being defined. This field is used for aggregation sets that are being fed by other aggregation sets in the database. If you select None, the aggregation set receives data directly from the data collection probes and is not fed by another aggregation set. If you select an aggregation set from the list, the new aggregation set will be fed from data of the specified aggregation set.

  The aggregation sets that appear in the Source Agg Set list box are those sets that follow the rules for feeding aggregation sets. For example, the granularity of the source aggregation set must be less than or equal to and an even multiple of the granularity of the aggregation set you are defining. The date consolidation must also be the same, including the first day of the week and the days to exclude. Day can feed week and month. Week cannot feed month. Month cannot feed week. See 6.2.4 for more information on feeding aggregation sets.

  Note: *When you create an aggregation set that is fed by another aggregation set, you must execute the* ***tqagg*** *utility program so that data is retrieved from the source aggregation set, consolidated, and stored in the fed aggregation set. See the 6.4.7 for information on the* ***tqagg*** *utility.*

Retain Data For
: specifies the length of time that data for this aggregation set should be retained in the database. Select retention time for the new aggregation set from the options in this list box.

  The maximum retention period that can be selected is dependent upon the granularity of the aggregation set being defined, based on a maximum of 31,680 data points that can be stored in one day for any aggregation set. Some common granularities and their maximum retention periods that can be selected are

Granularity	**Maximum Retention Period**
15 seconds	5 days
30 seconds	11 days
1 minute	22 days
5 minutes	110 days
10 minutes	220 days
30 minutes	660 days
1 hour	1320 days

Expanded Agg Set
: allows you to specify that you want your aggregation set to be expanded instead of using the default compact aggregation set. An expanded aggregation set provides more complete reporting of standard deviations than with compact aggregation sets along with increased accuracy for very large or very small data point values. The advantage of compact aggregation sets is that they require much less disk space to store the data while still providing accurate data point values.

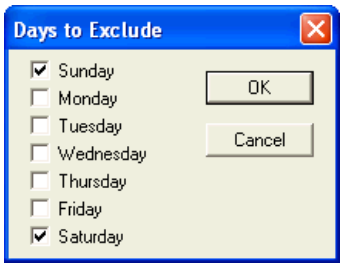
Date Consolidation

The Date Consolidation box contains a Type list box, a First Day list box, and an Exclude button for specifying attributes for consolidating the data points in your data across dates. Date consolidation determines if dates are consolidated into a sample interval.

Type list box
: specifies the group type for the consolidation. The available options are Day, Week, and Month. Day indicates that no consolidation of data points from other days is done. Week indicates that all data points from a week for each sample interval will be combined into one point. Month indicates that all data points from a month for each sample interval will be combined into one data point. For example, if the Type is Week, the data point for the 08:00 sample interval is the average value for each day of that week at 08:00. If the Type is Month, the data point for 08:00 for a specific month is the average of all values from the 08:00 sample interval for each day of the month.

First Day list box
: specifies the day of the week that should be considered the first day of the week. This list box is only available when the Type is Week.

Exclude button
: specifies the days of the week to exclude from data consolidation. This button is only available when the Type is Week or Month. When you click the Exclude button, the Days to Exclude dialog box is displayed.



When you check any of the boxes, the host software excludes the data from those days when consolidating data.

Data Collection Times

The Data Collection Times box contains a Range and an Interval scroll bar to make the time specifications for the data in the aggregation set.

Range scroll bar
: adjusts the time range of your data, that is the start and end times for data collection. Drag the scroll box or use the arrows on the scroll bar to specify the time range of your data. The time range adjusts at the rate specified by the Interval scroll bar. For example, if the Interval is 4 hours, the time range will be a period of 4 hours (08:00 to 12:00, 09:00 to 13:00, and so on). If the time range covers a period that crosses midnight, an indicator appears next to the end time stating that the end time is on the next day.

Interval scroll bar
: allows you to adjust the time interval specified for the time range. Drag the scroll box or use the arrows on the scroll bar to increase or decrease the size of the time interval. The interval adjusts at the rate of the point size shown in the Granularity drop-down list box. If the granularity is less than 5 minutes, 5 minutes is used.

When your definition is complete, click OK. The new aggregation set name is inserted into the Aggregation Sets list box in the Aggregation Set Maintenance dialog box (see 6.3.1). To define the aggregation set to the host software, you must click OK in the Aggregation Set Maintenance dialog box.

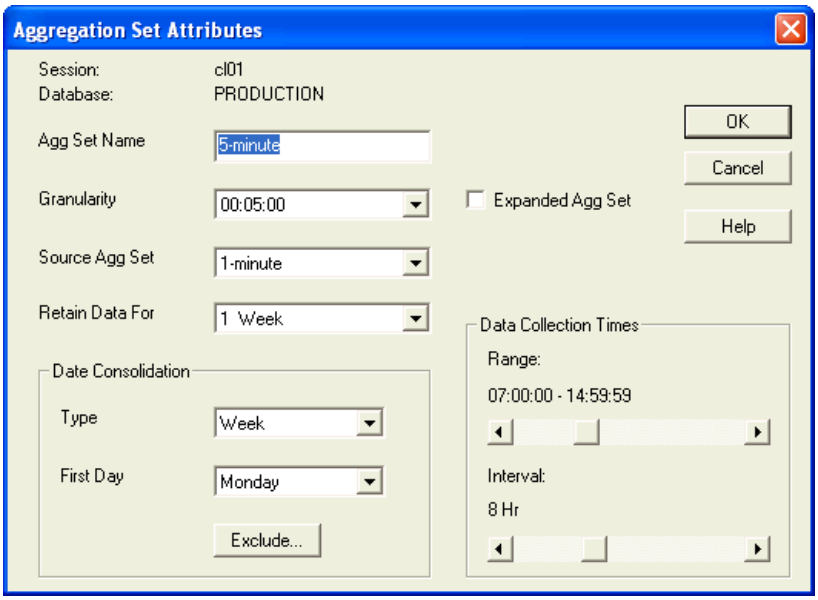
Note: *When you create an aggregation set, a new database file is created.*

6.3.3. Displaying or Modifying Aggregation Set Definitions

You can modify an aggregation set definition before that definition has been sent to the host software. If an aggregation set has been defined to the host, you cannot modify the definition. You can only display the aggregation set definition.

Note: *To modify an aggregation set after it has been defined to the host, you can create a new aggregation set definition (see 6.3.2) with the necessary attributes and then, optionally, delete the incorrect aggregation set (see 6.3.4).*

To display an aggregation set or modify an aggregation set definition before it has been defined to the host, select an aggregation set from the Aggregation Sets list box in the Aggregation Set Maintenance dialog box (see 6.3.1) and click Open. You can also double-click on the aggregation set name in the Aggregation Sets list box. The Aggregation Set Attributes dialog box is displayed.



Make your changes, as required. For a description of the fields in this dialog box, see 6.3.2.

When you click OK, your changes are accepted and you return to the Aggregation Set Maintenance dialog box (see 6.3.1). Your changes are not sent to the host software until you click OK in the Aggregation Set Maintenance dialog box (see 6.3.1).

6.3.4. Deleting an Aggregation Set

To delete an aggregation set from a database, select an aggregation set name from the Aggregation Sets list box in the Aggregation Set Maintenance dialog box (see 6.3.1). When you click the Delete button, an alert box verifies your request to delete the aggregation set.

When you click Yes in this alert box, the aggregation set name is removed from the Aggregation Sets list box in the Aggregation Set Maintenance dialog box (see 6.3.1). To delete the aggregation set definition from the database, you must click OK in the Aggregation Set Maintenance dialog box.

You can cancel the deletion of an aggregation set either by clicking the No button in the alert box or by clicking Cancel in the Aggregation Set Maintenance dialog box (see 6.3.1).

Note: *When you remove an aggregation set, the aggregation set file is deleted from the database*

6.4. Maintaining Aggregation Sets with tqasa

The **tqasa** program provides an optional method for performing the following tasks associated with maintaining aggregation sets:

- Create aggregation sets (see 6.4.1)
- Remove an aggregation set that is no longer needed from the database (see 6.4.2)
- Display information about the aggregation sets defined for a database (see 6.4.3)
- Unload aggregation set definitions in a database (see 6.4.4)
- Load aggregation set definitions in a database (see 6.4.5)
- Convert an existing expanded aggregation set into compact format (see 6.4.6)

After you create, remove, or change an aggregation set, you must customize your data collection to take advantage of any aggregation changes (see 6.4.7).

Note: *The capability to maintain multiple aggregation sets is available only with TeamQuest Baseline. TeamQuest Online allows only one aggregation set in the database.*

After you create, remove, or change an aggregation set, you may need to update the TeamQuest Baseline probe runstreams in SYSLIB*RUN$ or the tq-ini element in SYSLIB*TQ-BASELINE. If you want the probes to use the new aggregation sets, you will need to update the runstreams or the tq-ini element. However, sometimes you might create an aggregation set that is fed by another aggregation set. This aggregation set will automatically be handled by the **tqagg** program without requiring any updates to the runstreams. Also, an aggregation set may be created only to load data from another system. In that case, you do not want to update the probe runstreams in SYSLIB*RUN$ or the tq-ini element.

6.4.1. Creating an Aggregation Set

You can use the **tqasa** program to create an aggregation set for a database. When you create an aggregation set, a new database file is created and the **tqasa** program displays the name of the file.

Note: *When you create an aggregation set that contains data consolidated from another aggregation set, you must run the* ***tqagg*** *program to consolidate this data (see 6.4.7).*

To create an aggregation set, the **tqasa** processor call has the following format:

```
@tqasa -i,aggset, [-d,databasename,] ;
       [-g,granularity,] [-s,starttime,] [-e,endtime,] ;
       [-t,grouptype,] [-k,keepcount,] [-f,fed-by,] ;
       [-x,exclude,] [-w,weekstart,] [-p]
```

where:

-i specifies the aggregation set to create in the database. The *aggset* parameter is the name of the new aggregation set.

-d specifies the database for which you want to create an aggregation set. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-g specifies the granularity for the aggregation set. The *granularity* parameter is expressed as *n*, *n*sec, *n*min, or *n*hour where *n* is the number of seconds, minutes, or hours. Do not enter a space between the number and the unit specification. If no unit is specified, seconds is assumed. The default is 30 minutes.

-s specifies the start time of the aggregation set. The *starttime* parameter is in the form of *hhmmss*. The default is 000000.

-e specifies the end time of the aggregation set. The *endtime* parameter is in the form of *hhmmss*. The default is 235959.

The time interval specified by the *starttime* and *endtime* parameters cannot exceed 24 hours. However, you can specify an aggregation set that starts at 070000 and ends at 305959, thus reflecting a time interval beginning at 7:00:00 a.m. and ending at 6:59:59 a.m. the next day.

-t specifies the group type of the aggregation set. Valid values for the *grouptype* parameter are day, week, and month. If no group type is specified, day is assumed.

-k specifies the amount of data that is kept for the aggregation set. The *keepcount* parameter is expressed as *n*, *n*min, or *n*hour where *n* is the number of minutes, hours, or groups for which the data is stored in the database. Do not enter a space between the number and the unit. If the *keepcount* parameter is not specified, 180 groups is assumed.

The maximum retention period that can be selected is dependent upon the granularity of the aggregation set being defined, based on a maximum of 31,680 data points that can be stored in 1 day for any aggregation set. Some common granularities and their maximum retention periods that can be selected are

Granularity	**Maximum Retention Period**
15 seconds	5 days
30 seconds	11 days
1 minute	22 days
5 minutes	110 days
10 minutes	220 days
30 minutes	660 days
1 hour	1320 days

-f specifies the aggregation set that feeds the aggregation set being created. The *fed-by* parameter is the name of an aggregation set that has already been defined for the database.

You must use the *fed-by* parameter if you want this aggregation set to receive data from another aggregation set in the database. If you do not specify the *fed-by* parameter, the aggregation set receives data only from the TeamQuest Baseline probes. Note that the TeamQuest Baseline probes support only day group types.

-x specifies the days of the week to exclude for the aggregation sets that have week and month group types specified. The *exclude* parameter has the format of *xxxxxxx* where *x* can be either 1 or 0. A value of 1 signifies a day to exclude. A value of 0 signifies a day to include.

This exclusion mask is a series of seven 1 and 0 digits with each digit representing a day of the week. The first digit represents Sunday, the second represents Monday, and so on through the remaining days of the week. The last digit represents Saturday.

For example, if you want your aggregation set to exclude Sunday and Saturday and to include all other days, the *exclude* parameter is specified as 1000001. If you want the aggregation set to include only Sunday and Saturday, the *exclude* parameter is specified as 0111110. If you do not specify the *exclude* parameter, 0000000 is assumed (that is, no days are excluded from the aggregation set).

-w specifies the day that is the first day of the week for an aggregation set with a group type of week. The *weekstart* parameter can be a digit from 0 to 6. The digits represent the days of the week as follows:

0 = Sunday
1 = Monday
2 = Tuesday
3 = Wednesday
4 = Thursday
5 = Friday
6 = Saturday

If you do not specify the *weekstart* parameter, 0 is assumed.

-p specifies that the aggregation set to be created is in expanded format. See 6.2.5 for a description of compact and expanded aggregation sets. If the -p option is not used, the aggregation set is created in compact format.

Example

The following example creates a compact aggregation set called 30-minute with a granularity of 30 minutes and a retention period of 35 days.

```
>@tqasa -i,30-minute,-g,30min,-k,35
>TQASA 7R4G (011312 11:04:45) Fri Jan 13 14:49:08 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>File created: TQDATA$*PROD30-minute
>
>Aggregation Set List:
>
>                                                              Max
>                                                              File
>     Name        Gran     Start      Stop       Keep         Size     Source Name
>------------  --------  --------  --------  ----------  ------------ ------------
>    1-minute  00:01:00  00:00:00  23:59:59    08:00:00     ---------
>
>   30-minute  00:30:00  00:00:00  23:59:59     35 days     ---------
>
>
>An '*' following Name indicates an expanded aggregation set.
>
>End TQASA
```

6.4.2. Removing an Aggregation Set

You can use the **tqasa** program to remove an existing aggregation set from a database. When you remove an aggregation set, the aggregation set file is deleted from the database.

To remove an aggregation set, the **tqasa** processor call has the following format:

```
@tqasa [-d,databasename,] -r,aggset
```

where:

-d specifies the database for which you want to remove an aggregation set. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-r specifies the name of the aggregation set to remove. The *aggset* parameter is the name of an existing aggregation set.

Note: *An aggregation set whose name is greater than 12 characters or that contains an uppercase letter must be deleted using TeamQuest Performance Software Administration.*

Example

The following example removes the 30-minute aggregation set from the default database. Any remaining aggregation sets are listed for your information.

```
>@tqasa -r,30-minute
>TQASA 7R4G (011312 11:04:45) Fri Jan 13 14:59:08 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>Aggregation set 30-minute removed.
>
>Aggregation Set List:
>
>                                                                  Max
>                                                                  File
>     Name        Gran     Start      Stop       Keep          Size     Source Name
>------------  --------  --------  --------  ----------  ------------ ------------
>    1-minute  00:01:00  00:00:00  23:59:59    08:00:00     23024 Trk
>
>
>
>An '*' following Name indicates an expanded aggregation set.
>
>End TQASA
```

6.4.3. Listing Aggregation Sets

The **tqasa** program allows you to produce a list of the aggregation sets currently defined for a database. The **tqasa** program displays the aggregation set name, granularity, start time, end time, keep count, maximum file size, and the source name of the aggregation set.

To list the existing aggregation sets for a database, the **tqasa** processor call has the following format:

```
@tqasa [-d,databasename]
```

where -d specifies the database for which you want to list the existing aggregation sets. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

Example

The following example shows how to list the aggregation sets for the default database.

```
>@tqasa
>TQASA 7R4G (011312 11:04:45) Fri Jan 13 16:22:10 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>Aggregation Set List:
>
>                                                              Max
>                                                              File
>     Name        Gran     Start      Stop       Keep         Size     Source Name
>-------------  --------  --------  --------  ----------  ------------ -----------
>     1-minute  00:01:00  00:00:00  23:59:59    08:00:00     23034 Trk
>    10-minute  00:10:00  00:00:00  23:59:59      8 days     13818 Trk
>       1-hour  01:00:00  00:00:00  23:59:59     35 days      8396 Trk  10-minute
>       8-hour  08:00:00  00:00:00  23:59:59    400 days     11996 Trk     1-hour
>
>An '*' following Name indicates an expanded aggregation set.
>
>
>End TQASA
```

6.4.4. Unloading Aggregation Set Definitions

The **tqasa** program allows you to export all of the aggregation set definitions currently contained in the performance database into a file. You can use this file to view the aggregation set definitions and update them by using a text editor. You can then reload the aggregation set definitions by using the -l option on the **tqasa** processor call (see 6.4.5).

To unload aggregation set definitions with **tqasa**, use the following processor call format:

```
@tqasa [-d,databasename,] -u,aggset-file
```

where:

-d specifies the database for which you want to unload aggregation set definitions. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-u specifies that an aggregation set unload operation is to be performed by **tqasa**. All aggregation set definitions in the database are exported into the file specified by the *aggset-file* parameter.

Example

The following shows an example of unloading aggregation set definitions from the default database into the file aggsetfile.

```
>@tqasa -u,aggsetfile
>TQASA 7R4G (011312 11:17:30) Fri Jan 13 08:17:27 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>...unloading aggregation set definitions to: aggsetfile
>2 definitions were unloaded.
>
>
>End TQASA
```

Aggregation Set Definition File Format

The aggregation set definitions appear in the following format in the output file created by **tqasa**. If you choose to edit this file, you should maintain this file format.

```
BEGIN AGGSETS unloaded on Fri Jan 13 08:17:28 2012                A
Name=1-minute                                                     B
Granularity=60
Start_Time=000000
Stop_Time=235959
Group_Type=DAY
Week_Start=SUNDAY
Exclude_Days=
Compact=YES
Keep_Count=8HOUR
Fed_By=
#                                                                 C
Name=10-minute                                                    D
Granularity=600
Start_Time=000000
Stop_Time=235959
Group_Type=DAY
Week_Start=SUNDAY
Exclude_Days=
Compact=YES
Keep_Count=8
Fed_By=
#
END AGGSETS                                                       E
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A The first line of the aggregation set definition file must be the header line BEGIN AGGSETS. It is followed by a series of aggregation set definitions.

B This line begins the first aggregation set definition. Each aggregation set definition is made up of 10 lines. The lines must appear in order and the label must appear exactly as shown followed by an equals (=) sign. The lines are as follows:

Name	The aggregation set name. A value for this field is required.
Granularity	The granularity of the aggregation set specified in seconds. For example, if the aggregation set contains hourly data, the granularity is 3600. A value for this field is required.
Start_Time	The time to start data collection for the aggregation set. It is specified as *hhmmss*. If no value is specified for this field, 000000 is used by default.
Stop_Time	The time to stop data collection for this aggregation set. It is specified as *hhmmss*. If no value is specified for this field, 235959 is used by default

Group_Type	The group type of the aggregation set. This value can be DAY, WEEK, or MONTH. If no value is specified for this field, DAY is used by default.
Week_Start	The day that is considered the first day of the week for the aggregation set. This value can be SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, or SATURDAY. A value is allowed for this field only if the Group_Type is WEEK. If no value is specified, SUNDAY is used by default.
Exclude_Days	The days of the week to exclude from data collection for the aggregation set. This list of values can include any of the following: SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, or SATURDAY. The values are separated by a space or a comma. For example, to exclude Sunday and Saturday from data collection, you would specify: Exclude_Days=SUNDAY,SATURDAY A value is allowed for this field only if Group_Type is WEEK. If no value is specified, no days are excluded by default.
Compact	Indicates whether or not this is a compact aggregation set. If it is a compact aggregation set, specify YES. Otherwise, specify NO. If no value is specified, YES is used by default.
Keep_Count	The amount of data that is kept for the aggregation set. It is expressed as *n*, *n*MIN, or *n*HOUR where *n* is the number of groups, minutes, or hours for which data is retained. If only *n* is specified, the length of a group is defined by Group_Type. Do not put a space between the number and the unit. A value for this field is required.
Fed_By	The name of the aggregation set that feeds the aggregation set being created. The specified aggregation set must already exist in the database. You must use the Fed_By field if you want this aggregation set to receive data from another aggregation set in the database. If you do not specify a value, the aggregation set receives data only from TeamQuest Baseline probes. Note that TeamQuest Baseline probes only support Group_Types of DAY.

C Comment lines begin with a # character.

D This line begins the second aggregation set definition.

E The last line of the file must be the END AGGSETS line.

6.4.5. Loading Aggregation Set Definitions

You can reload the performance database with a new set of aggregation set definitions by using the -l option on the **tqasa** processor call. You should use the file as created by the **tqasa** -u option as input for loading aggregation set definitions (see 6.4.4). If you update the aggregation set definition file, you should ensure that the format of the file remains consistent with what was output by the **tqasa** -u option.

Existing aggregation set definitions cannot be updated by loading a new set of definitions. If the load process encounters an existing aggregation set, a warning message is printed and that definition is ignored. If you want to replace your existing aggregation sets with the ones in the load file, use the -r,ALLSETS option with the -l option. This will cause all existing aggregation sets to be deleted before the new ones are loaded.

Note: *Use of the -r,ALLSETS option will cause all existing aggregation sets to be deleted from your database. Both the aggregation set definition and all data contained in the aggregation set are deleted. When aggregation sets are loaded, only the aggregation set definition is created in the database. No data is contained in the loaded aggregation sets until either the TeamQuest Baseline probes or the* ***tqagg*** *program store data into the aggregation set.*

To load aggregation set definitions with **tqasa**, use the following processor call format:

```
@tqasa [-d,databasename,] -l,aggset-file, [-r,ALLSETS]
```

where:

-d — specifies the database for which you want to load aggregation set definitions. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-l — specifies that an aggregation set load operation is to be performed by **tqasa**. All aggregation set definitions in the file specified by the *aggset-file* parameter are imported into the database.

-r,ALLSETS — deletes all existing aggregation sets before loading the new ones. If you do not specify this option, no existing aggregation sets are deleted.

Example

The following shows an example of loading aggregation set definitions contained in the file aggsetfile. Because the -r,ALLSETS option is specified, all existing aggregation sets will be deleted before the new ones are loaded.

```
>@tqasa -l,aggsetfile,-r,ALLSETS
>TQASA 7R4G (011312 11:17:30) Fri Jan 13 08:17:27 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>...removing all existing aggregation sets
>...loading aggregation set definitions from: aggsetfile
>File created:  TQDATA$*PROD1-minute
>File created:  TQDATA$*PROD10-minute
>2 definitions were loaded.
>
>Aggregation Set List:
>
>                                                                  Max
>                                                                  File
>     Name        Gran     Start      Stop       Keep         Size     Source Name
>-------------  --------  --------  --------  ----------  ------------ -----------
>     1-minute  00:01:00  00:00:00  23:59:59    08:00:00     ---------
>    10-minute  00:10:00  00:00:00  23:59:59      8 days     ---------
>
>
>An '*' following Name indicates an expanded aggregation set.
>
>End TQASA
```

6.4.6. Converting Expanded Aggregation Sets

The **tqasa** program allows you to convert an existing expanded aggregation set into compact format. It is recommended that you back up your performance database before performing this conversion. Also, all processes that are currently using the selected database must be terminated before running **tqasa** to convert an aggregation set.

To convert an existing aggregation set from expanded to compact format, the **tqasa** processor call has the following format:

```
@tqasa -c[,-d,databasename]
```

where:

-c calls the **tqasa** program to begin aggregation set compaction.

-d specifies the database that contains the aggregation set you want to convert. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

Example

The following example shows how to convert an aggregation set from expanded to compact format:

```
>@tqasa -c                                                              A
>TQASA 7R4G (011312 11:04:45) Fri Jan 13 16:22:10 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>*******************************************************              B
>*Aggregation set compaction has been selected
>*for database <PRODUCTION>.
>*
>*It is recommended that you backup your database
>*before continuing with this operation.
>*
>*All processes currently using the selected
>*database must be stopped before continuing.
>*******************************************************
>Do you want to continue (y,<n>)?                                       C
>y
>Enter the aggregation set name:                                        D
>1-minute
>
>Compaction is complete for aggregation set 1-minute.                   E
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A Call the **tqasa** program with -c option to convert your aggregation sets.

B The **tqasa** program displays messages for your verification.

C If you enter **y** at the prompt, the **tqasa** program will continue and prompt you for the name of the aggregation set to convert.

Enter **n** if you have not backed up your database or if there are processes currently running that are using the selected database.

D Enter the name of the aggregation set that you want to convert from expanded to compact format.

E The **tqasa** program performs the conversion of the aggregation set and displays a message when the conversion is complete.

6.4.7. Performing Data Aggregation

When you create an aggregation set that is fed by another aggregation set, you must execute the **tqagg** program so that data is retrieved from the source aggregation set, consolidated, and stored into the fed aggregation set. The **tqagg** program performs data aggregations for all fed aggregation sets defined in a performance database. The **tqagg** program is also responsible for consolidating and removing table records. The runstream SYSLIB*RUN$.tqagg is started to run **tqagg**. By default, this runstream is set up to run at midnight.

The **tqagg** program has the following processor call format:

```
@tqagg [-d,databasename,] [-c,] [-t,hhmm,] [-r,days]
       [-s,aggsetname, -f,aggsetname]
```

where:

-d specifies the database for which you want to perform data consolidation. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-c causes the **tqagg** program to run in continuous mode. In continuous mode, **tqagg** performs the data consolidation each day at midnight. If you want to run **tqagg** in continuous mode, you should update the **tqagg** runstream appropriately.

If the -c option or -t option is not specified, **tqagg** performs data consolidation for each fed aggregation set and then terminates.

-t causes the **tqagg** program to run in continuous mode. This option differs from the -c option in that you can specify the time you want the data consolidation to take place each day. The time is specified as *hhmm*. If you want **tqagg** to run in continuous mode, you should update the **tqagg** runstream appropriately.

If the -c option or -t option is not specified, **tqagg** performs data consolidation for each fed aggregation set and then terminates.

-r allows you to reprocess a time period that has been consolidated previously.

With the -r option, you can specify the number of days back to begin the consolidation process. For example, if you specify 2 days, the **tqagg** program starts aggregating the data beginning two days prior to the last point it had consolidated. Previously consolidated data points remain unchanged, but any data points stored to the source aggregation set since the last **tqagg** execution are now consolidated.

Without the -r option, **tqagg** begins the data consolidation for an aggregation set one day prior to the last point it had previously consolidated. It continues to consolidate data until it reaches the end of the data in the source aggregation set.

Using the -r option allows you to repopulate an aggregation set after a change in the content of the source aggregation set. This may be necessary if data is loaded into the database for a time period that had already been processed by **tqagg**.

The default action of **tqagg** is to reprocess the data points from the previous day at each execution. This is necessary to ensure that the data collected by the probes running in offline mode is consolidated (that is, **tqsys**, **tqmpr**, **tqtlp**, **tqtpm**, **tqrtpm**, and **tqiot**). If **tqagg** starts before the offline probes are finished for the day, some of the data stored by the offline probes would not be aggregated until the next day when **tqagg** is run.

If you do not use any of the offline probes or can ensure that **tqagg** does not run until after all of the offline probes are complete, you can inhibit any reprocessing of data by specifying the -r option with 0 days. When 0 days is specified, **tqagg** starts aggregating the data immediately following the last point it had consolidated.

-s allows you to set the source aggregation set.

-f allows you to set the destination aggregation set.

Notes:

- *The -r option is only required in special cases. Normally, you should not need to use the -r option.*
- *By default, the SYSLIB*RUN$.TQAGG runstream is started automatically using SYSLIB*RUN$.TQCTL. The tqagg runstream is started with a start time of midnight. The tqagg run remains in backlog until midnight. After it finishes processing, tqagg puts itself into backlog again until the next midnight. The -c and -t options provide an alternative method of controlling when tqagg is run.*

Section 7
Customizing Alarms and Thresholds

Note: *The alarms and thresholds capability is available with TeamQuest Baseline only.*

7.1. Introduction

The **tqzip** program is used to administer the alarms and thresholds capability of the performance database. Default thresholds are supplied and can be loaded into your performance database (see 3.3.2). If you are content with the default threshold values, no additional administration is required. However, if you want to customize your thresholds, you can perform the following tasks:

- Define thresholds (see 7.4, 7.5, and 7.6)
- Define thresholds for derived statistics (see 7.8 and 7.9)
- View threshold information (see 7.7 and 7.10)
- Delete thresholds (see 7.11)
- Notify the probes of changes in thresholds (see 7.12)
- Unload thresholds (see 7.13)
- Load thresholds (see 7.14)
- Turn on and off threshold checking (see 7.15)
- Create a file containing alarm messages (see 7.16)

7.2. Concepts

The TeamQuest Baseline probes collect data from various data sources at the sample interval defined for the user-specified aggregation set. The probes can also check the data obtained at each sample interval to determine if the data values exceed user-specified threshold limits. If a data value for a parameter is found to exceed a threshold, the probe can generate an alarm to alert you that a threshold condition has been met.

To determine when a threshold condition has been met, the TeamQuest Baseline probes use the concept of threshold zones. You define threshold information for a parameter, including the number of zones, the normal zone, and the number of seconds before the alarm is triggered. The probe generates an alarm whenever the value for a given parameter is found to be out of the current zone for the specified number of seconds.

The following terms are important to the understanding of threshold zones:

- The *number of zones* defines the number of threshold zones in use for a given parameter. It is possible to have either two or three threshold zones for a parameter.
- The *normal zone* is the zone that the parameter value normally falls into.
- The *current zone* is the zone that contained the last value for the parameter. When an alarm is generated, the current zone is changed to be the zone that contains the current value for the parameter. The current zone is initially set to the normal zone.
- The *seconds before alarm* specifies the number of seconds that the parameter value must be out of the current zone before an alarm is raised. Since threshold checking is done at each sample, the number of seconds specified is actually converted into the number of samples.

The **tqzip** program is used to define threshold information for a parameter. When defining thresholds for a parameter, you can define either two or three threshold zones. If you have two threshold zones, you specify only one threshold value. If you have three threshold zones, you specify two threshold values. Each threshold value is the point that separates one threshold zone from another.

Alarm messages generated by TeamQuest Baseline include a severity indicator. The following severities are used for the parameters with two threshold zones:

NORMAL
: The value for the parameter has returned to the normal zone.

CRITICAL
: The value for the parameter has crossed the threshold leaving the normal zone.

The following severities are used for parameters with three threshold zones:

NORMAL
: The value for the parameter has returned to the normal zone.

WARNING
: The normal zone is zone 1 or zone 3 and the value for the parameter has moved one zone away from the normal zone.

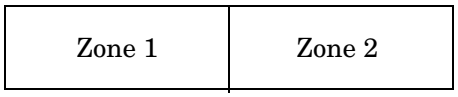
CRITICAL
: The value for the parameter has moved two zones away from the normal zone.

  or

  The normal zone is zone 2 and the value for the parameter has moved one zone away from the normal zone.

Figure 7–1 illustrates the concept of threshold zones. At each data sample, the probe performs threshold checking on those parameters that have threshold definitions. The probe first determines the zone in which the current value of the parameter is found. The zone is determined as follows (threshold value 1 is t1, threshold value 2 is t2):

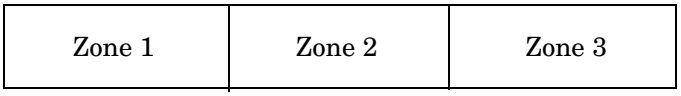
2 Zones Defined



Values < t1 are in Zone 1

Values > or = t1 are in Zone 2

3 Zones Defined



If t1 < t2
Values (> or = t2) are in Zone 3
Values (> or = t1) and (< t2) are in Zone 2
Values (< t1) are in Zone 1

If t1 > t2
Values (> or = t1) are in Zone 1
Values (< t1) and (> or =t2) are in Zone 2
Values (< t2) are in Zone 3

Figure 7–1. Threshold Zones

The following example illustrates the usage of alarms and thresholds on a system. You might want a message displayed on the console if demand response time exceeds 5 seconds for a duration of at least 60 seconds. You would define two zones where Zone 1 is the normal zone and threshold value 1 (t1) has a value of 5 seconds. The following situations could occur:

- If demand response time is 3 seconds, the parameter is in Zone 1 and no alarm is issued.
- If demand response time increases to 10 seconds (the parameter moves to Zone 2), then remains at this level for 40 seconds and decreases to 3 seconds (the parameter moves to Zone 1), no alarm is issued.

  If demand response time increases to 10 seconds (Zone 2) and remains at this level for 60 seconds, an alarm message with a severity of CRITICAL is sent to the console. The current zone for demand response time is changed to be Zone 2. The next alarm would be generated when the value for demand response time returned to Zone 1 for at least 60 seconds. This alarm would have a severity of NORMAL indicating a return to the normal zone (that is, zone 1).

7.3. Opening the Alarms and Thresholds Menu

The **tqzip** program displays a series of menus for you to select the tasks you want to perform. To perform maintenance on alarm and threshold information, you execute the **tqzip** program and select Alarms and Thresholds from the Administration Menu.

```
>@tqzip
>TQZIP 7R4G (011312 11:04:45) Fri Jan 13 09:19:55 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>
>Using database TQDATA:PRODUCTION
>
>
>     Administration Menu
>
>    1)  Alarms and Thresholds
>    2)  Tables
>
>   99)  Exit TQZIP
>
>
>Enter your selection:
>1
```

The **tqzip** program then displays the Alarms and Thresholds Menu.

```
>      Alarms and Thresholds Menu
>
>    1)  Define Thresholds
>    2)  View Thresholds
>    3)  Delete Thresholds
>    4)  Notify Probes of Threshold Changes
>    5)  Turn Threshold Checking On/Off
>    6)  Generate Alarm List
>
>   99)  Exit Alarms and Thresholds Menu
>
>Enter your selection:
>
```

To perform a maintenance task, enter its associated number at the prompt. To return to the Administration Menu, enter **99**.

When you select the various maintenance tasks for the Alarms and Thresholds Menu, the **tqzip** program prompts you to supply values or other information. Default values are displayed for each prompt in angle brackets (<>). To use a default, press the transmit key without entering a value.

7.4. Creating Threshold Definitions

To perform threshold checking on a given parameter or derived statistic in the performance database, you must create a threshold definition for the parameter or derived statistic.

To create a threshold definition, select Define Thresholds from the Alarms and Threshold Menu (see 7.3) by entering **1** at the prompt. The **tqzip** program displays prompts as shown in the following example.

```
>Enter your selection :
>1
>
>Create threshold, Update threshold, or Exit (c,u,<e>)?                A
>c
>
>System Name                                                           B
>1) $DERIVED$
>2) CLO1
>3) CLA
>
>Enter number, name, <blank> for more, or E to exit :
>2
>Statistic Name                                                        C
>199) Output message requeues
>200) Output messages
>201) Outputs queued to CMS
>202) Page I/O rate
>203) Page que
>204) Pass-off input message
>205) Pass-off messages
>206) Percent Busy
>207) Percent Load Fail
>208) Percent Reqs Queued
>209) Percent Reqs Read
>210) Privileged Instruction
>211) Program File Loads/Sec
>212) QPR active requests
>213) QPR inputs queued
>214) QPR max active requests
>215) QPR spooled requests
>216) QPR waiting requests
>
>Enter number, name, <blank> for more, or E to exit :
>206
>
>Full Statistic Name                                                   D
>1) IO Activity:Control Unit:Summary:Percent Busy
>2) IO Activity:Disk:Summary:Percent Busy
>3) IO Activity:Logical Channel:Summary:Percent Busy
>4) IO Activity:Other:Summary:Percent Busy
>5) IO Activity:Tape:Summary:Percent Busy
>
```

```
>Enter number, name, <blank> for more, or E to exit :
>2
>
>Resource Name                                                              E
>1) DSKA0
>2) DSKA1
>3) DSKB0
>4) DSKB1
>5) DSKC0
>6) DSKC1
>7) DSKD0
>8) DSKD1
>9) DSKF0
>10) DSKF1
>11) DSKF2
>12) DSKF3
>13) DSKF4
>14) DSKF5
>15) DSKF6
>16) DSKF7
>17) DSKK0
>18) DSKK1
>
>Enter number, name, <blank> for more, or E to exit :
>10
>
>CL01:IO Activity:Disk:Summary:Percent Busy:DSKF1:::                        F
>Enable threshold checking (y,<n>)?                                         G
>y
>Number of threshold zones (<2>) :                                          H
>3
>Value is usually in zone (<1>) :                                           I
>1
>First threshold value (<00000.00>) :                                       J
>50
>Second threshold value (<00000.00>)) :                                     K
>75
>Value must be in new zone for <0000> seconds before raising alarm :        L
>60
>Disable threshold checking for <0000> seconds after raising alarm :        M
>60
>
>To clear a text string, enter a period at the prompt.
>
>Alarm identifier (<>) :                                                    N
>Percent Busy
>User text line 1 (<>) :                                                    O
>When more than 50% busy, call John at ext 2332
>User text line 2 (<>) :
>
>User text line 3 (<>) :
>
>
```

```
>CL01:IO Activity:Disk:Summary:Percent Busy:DSKF1:::                    P
>1. Threshold checking : ON
>2. Threshold 1        : 00050.00 Threshold 2     : 00075.00
>3. Number of zones    : 3        Normal Zone     : 1
>4. Secs before alarm  : 0060     Secs delay      : 0060
>5. Alarm identifier   : Percent Busy
>6. User text : When more than 50% busy, call John at ext 2332
>7. User text :
>8. User text :
>
>Save this entry (<y>,n) :
>y
>
>Do you want to define another threshold (y,<n>)?                       Q
>n
>
>Threshold checking is ON.                                              R
>Notify active probes of these threshold changes (y,<n>)?
>y
>
>Probe notification complete.
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A Enter **c** at the prompt to create a new threshold definition.

The **tqzip** program then displays a series of prompts for you to select the parameter or derived statistic name for which you want to define thresholds.

B If more than one system exists in the database, the **tqzip** program displays a list of the systems in the database. The name $DERIVED$ is selected to define thresholds for any of the derived statistics in your database. See 7.8 for information on creating derived statistic thresholds.

At the prompt, enter the system for which the threshold definition will be created. You can enter either the number or the system name.

C The **tqzip** program displays a list of the statistics that are associated with the system name you chose in B.

At the prompt, select the statistic name for which you want to create a threshold definition. You can enter either the number of the statistic or the statistic name.

D If the statistic name you chose in C is used in multiple database parameters, the **tqzip** program displays a list of full statistic names.

At the prompt, select the statistic name that fully identifies the statistic for which you want to define thresholds. You can enter either the number of the statistic or the statistic name.

E If the statistic you have selected in D is associated with a resource, workload, or application group, the **tqzip** program prompts you for the required values to complete the parameter name selection. This example shows the prompts for a resource name selection.

At the prompt, enter the resource for the statistic. You can enter either the number of the resource or the resource name.

F When the parameter has been selected, the **tqzip** program displays the full name of the parameter. It then displays a series of threshold definition prompts.

G You can enable or disable individual threshold definitions. Enter **y** at the prompt to enable threshold checking for the parameter. Enter **n** at the prompt to disable threshold checking for the parameter.

You can create a threshold definition for a parameter even if you enter an **n** at this prompt. The threshold definition will exist in the performance database, but will not be used until you update it to enable threshold checking by entering a **y** at the prompt.

H Enter the number of threshold zones for the parameter. You can define either 2 or 3 threshold zones for a parameter. (See 2.5 for more information on threshold zones.)

I Enter the number of the normal zone for a sampled value for this parameter. This is used to determine if a zone change has occurred on the first sample taken by a probe.

J Enter the first threshold value for the parameter. This value is used to determine the boundary between threshold zone 1 and threshold zone 2.

K If you are defining three threshold zones, enter the second threshold value for the parameter. This value is used to determine the boundary between threshold zone 2 and threshold zone 3.

L Specify the number of seconds that the current value for this parameter must be in a new zone before an alarm is raised. This value should be a multiple of the sample rate.

M Specify how long you want threshold checking disabled for this parameter after an alarm has been raised for it. This value should be a multiple of the sample rate. This value will not raise an additional alarm. An alarm will only be raised when a threshold level is crossed as the value goes up or down.

N Enter a text string that identifies the parameter for which the threshold is being defined. This string can be 35 characters or letters in length. This string is used for creating the alarm messages that are displayed at the system console.

O Enter the text strings providing site-specific information that are displayed when an alarm occurs for this parameter. You can enter up to three text strings. Each text string must be less than 61 characters in length due to console limitations. These lines are optional and, if present, are added as additional lines in the alarm message sent to the system console.

P The **tqzip** program displays the threshold definition for approval before it is actually written to the database. Then, it asks you if you want to save the new threshold definition.

If you enter **y** at this prompt, the threshold definition is saved in the performance database.

If you enter **n** at the prompt, the threshold definition is not saved in the performance database.

Q If you want to create or update another threshold definition, enter **y** at the prompt. The **tqzip** program returns you to the create or update threshold prompt in A.

If you have finished creating threshold definitions, enter **n** at the prompt.

R When any changes have been made to the threshold definitions in the performance database, the **tqzip** program displays a final prompt. If overall threshold checking is turned on, you must notify any active probes of the threshold definition changes before they will take effect. Enter **y** at the prompt to notify the probes. You can also use the Notify Probes of Threshold Changes selection on the Alarms and Thresholds Menu to perform this task (see 7.12).

If overall threshold checking is turned off, the **tqzip** program displays the following prompt:

```
>Threshold checking is OFF.
>Do you want to turn it ON (y,<n>)?
>y
>Threshold checking is now ON.
>*********************************************
>*   TQALM must be executing on your system     *
>*       when threshold checking is ON.         *
>*********************************************
```

Overall threshold checking must be turned on before any of the individual threshold definitions are checked for threshold violations. To turn on overall threshold checking, you can either answer **y** to this prompt or use the Turn Threshold Checking On/Off selection on the Alarms and Threshold Menu to perform this task.

When using threshold checking, you must also ensure that the **tqalm** program is executing on your system. See 7.17 for information on the **tqalm** program and alarm handling.

7.5. Reusing Threshold Definitions

When you are defining a threshold for a parameter that has a resource, workload, or application group associated with it, the **tqzip** program allows you to use the same threshold definition for other resources, workloads, or application groups. During the threshold definition sequence, the **tqzip** program displays the appropriate prompts, depending on the parameter. The following example shows the prompt for a resource parameter. These prompts are displayed before you are asked to create or update another threshold (see `Q` in 7.4).

```
>Define thresholds for another resource (<y>,n)?                          A
>y
>
>Resource Name
>1) DSKA0
>2) DSKA1
>3) DSKB0
>4) DSKB1
>5) DSKC0
>6) DSKC1
>7) DSKD0
>8) DSKD1
>9) DSKF0
>10) DSKF1
>11) DSKF2
>12) DSKF3
>13) DSKF4
>14) DSKF5
>15) DSKF6
>16) DSKF7
>17) DSKK0
>18) DSKK1
>
>Enter number, name, <blank> for more, or E to exit:                      B
>2
>
>
>CL01:IO Activity:Disk:Summary:Percent Busy:DSKA1:::                       C
>1. Threshold checking : ON
>2. Threshold 1        : 00050.00  Threshold 2      : 00075.00
>3. Number of zones    : 3         Normal Zone      : 1
>4. Secs before alarm  : 0060      Secs delay       : 0060
>5. Alarm identifier   : Percent Busy
>6. User text : When more than 50% busy, call John at ext 2332
>7. User text :
>8. User text :
>
>Modify the thresholds (y,<n>)?                                           D
>
>Threshold entry created.
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A To reuse a threshold definition for a resource, enter **y** at the prompt. The **tqzip** program returns you to the list of resources (see E in 7.4).

B Enter the new resource for which you want to define a threshold. You can enter either the number of the resource or the resource name.

C The **tqzip** program displays the threshold definition for the last resource defined.

D The **tqzip** program allows you to use the threshold definition as is or to modify it for the current resource.

If you enter **n** at the prompt, the new threshold definition is saved to the performance database immediately.

If you answer **y** at the prompt, the **tqzip** program displays the series of threshold definition prompts with the defaults set to those displayed in the definition (see G through P in 7.4).

7.6. Updating Threshold Definitions

The **tqzip** program allows you to update threshold definitions that are currently defined in the performance database. To update a threshold definition, select Define Thresholds from the Alarms and Threshold Menu (see 7.3) by entering **1** at the prompt. The **tqzip** program displays the following prompts to update thresholds for a parameter:

```
>Enter your selection :
>1
>
>Create threshold, Update threshold, or Exit (c,u,<e>)?                A
>u
>
>Current Threshold Definitions                                         B
>1) CL01:IO Activity:::Disk IOs/Sec::::
>2) CL01:IO Activity:Disk:Summary:IOs/Sec:DSKF1:::
>3) CL01:IO Activity:Disk:Summary:Percent Busy:DSKF1:::
>
>Enter number, name, <blank> for more, or E to exit :                  C
>3
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A To update a threshold, enter **u** at the prompt.

B The **tqzip** program displays a list of the parameters that currently have threshold definitions.

C Select the parameter for which you want update the threshold definition. You can enter either the number of the parameter or the full parameter name.

If the parameter list is longer than will fit on one screen, enter a blank line to view more of the list.

After you select a parameter name, the **tqzip** program displays the series of threshold definition prompts with the current value set as the default for each prompt (see G through P in 7.4). You can modify any of the fields. To use the default, press the transmit key without entering a value at the prompt.

7.7. Viewing Thresholds

The **tqzip** program allows you to view an existing threshold definition. Select View Thresholds from the Alarms and Thresholds Menu (see 7.3) by entering **2** at the prompt. The **tqzip** program then prompts you with a list of parameter names that currently have threshold definitions in the database as shown in the following example.

```
>Enter your selection:
>2
>
>Current Threshold Definitions                                          A
>1) CL01:IO Activity:::Disk IOs/Sec::::
>2) CL01:IO Activity:Disk:Summary:IOs/Sec:DSKF1:::
>3) CL01:IO Activity:Disk:Summary:Percent Busy:DSKF1:::
>
>Enter number, name, <blank> for more, or E to exit :                   B
>3
>
>CL01:IO Activity:Disk:Summary:Percent Busy:DSKF1:::                     C
> 1.  Threshold checking : ON
> 2.  Threshold 1        : 00050.00  Threshold 2      : 00075.00
> 3.  Number of zones    : 3         Normal zone      : 1
> 4.  Secs before alarm  : 0060      Secs delay       : 0060
> 5.  Alarm identifier   : Percent Busy
> 6.  User text : When more than 50% busy, call John at ext 2332
> 7.  User text :
> 8.  User text :
>
>
>Do you want to view another threshold (y,<n>)?                          D
>
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A The **tqzip** program displays the list of threshold definitions currently defined in the performance database.

B Select the parameter you want to view. You can enter either the number of the parameter or the full parameter name.

If the parameter list is longer than will fit on one screen, enter a blank line to view more of the list.

C After you select a parameter name, the **tqzip** program displays the threshold definition for that parameter.

D If you enter **y** at the final prompt, the **tqzip** program prompts you for another parameter name to view.

If you enter **n** at the prompt, the **tqzip** program returns to the Alarms and Thresholds Menu (see 7.3).

7.8. Creating Derived Statistic Threshold Definitions

To create a threshold definition for a derived statistic, select Define Thresholds on the Alarms and Threshold Menu (see 7.3) by entering **1** at the prompt. The **tqzip** program displays the following prompts to define thresholds for a derived statistic.

```
>Enter your selection :
>1
>
>Create threshold, Update threshold, or Exit (c,u,<e>)?              A
>c
>
>System Name                                                        B
>1) $DERIVED$
>2) venus
>
>Enter number, name, <blank> for more, or E to exit :
>1
>
>Statistic Name                                                     C
>1) Derived Statistic
>2) RespTimeGoal
>3) ResponseTime
>4) UserCpuUtil
>
>Enter number, name, <blank> for more, or E to exit :
>4
>
>Select the resource(s) to use for this derived statistic.          D
>
>Resource Name (precede choice with '-' to deselect)
```

```
>1)   IP000
>2)   IP001
>3)   IP002
>4)   IP003
>
>Enter number, name, <blank> for more, or E to exit :
>1
>Resource Name (precede choice with '-' to deselect)
>1) * IP000
>2)   IP001
>3)   IP002
>4)   IP003
>
>Enter number, name, <blank> for more, or E to exit :
>2
>
>Resource Name (precede choice with '-' to deselect)
>1) * IP000
>2) * IP001
>3)   IP002
>4)   IP003
>
>Enter number, name, <blank> for more, or E to exit:
>e                                                                    E
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A Enter **c** at the prompt to create a new threshold definition.

The **tqzip** program then displays a series of prompts for you to select the name of the parameter or derived statistic for which you want to define thresholds.

B The **tqzip** program displays a list of the system names in the database. To define thresholds for derived statistics, you must select $DERIVED$ from this list. You can enter either the number or the name $DERIVED$.

C The **tqzip** program displays a list of the derived statistics in the database. Select the derived statistic name for which you want to create a threshold definition. You can enter either the number of the statistic or the statistic name.

D A derived statistic may require qualifiers to calculate its values. Qualifiers can be system names, resource names, workload set/workload, or application group names. If the derived statistic you chose in C requires qualifiers, you are prompted for them now. This example shows a derived statistic that requires system and resource name qualifiers. A prompt did not appear for the system because there is only one system name other than $DERIVED$. If only one qualifier exists in the prompt list, it is automatically selected for you.

To add a selection to the qualifier list, you must enter the number or name of the qualifier value. To remove a selection from the list, you enter the number of the qualifier value preceded by a hyphen (-). An asterisk is used to indicate which qualifier values in the list have been selected. You continue adding or removing selections in the qualifier list until entering an E to exit.

The **tqalm** program uses these qualifier values to determine how threshold checking is to be done for a derived statistic. In this example, the value of UserCpuUtil for IP000 and IP001 will each be calculated and separately checked against the UserCpuUtil threshold definition for possible threshold violations. Threshold checking and alarm generation for IP000 and IP001 are independent of each other. However, any changes to the threshold definition for UserCpuUtil will affect both IP000 and IP001.

E After you have finished selecting the qualifier values for this derived statistic threshold, the **tqzip** program displays the series of threshold definition prompts (see G through P in 7.4). You can modify any of the fields.

7.9. Updating Derived Statistic Threshold Definitions

The **tqzip** program allows you to update threshold definitions that are currently defined in the performance database. To update a derived statistic threshold definition, select Define Thresholds on the Alarms and Threshold Menu (see 7.3) by entering **1** at the prompt. The **tqzip** program displays the following prompts to update thresholds for a derived statistic:

```
>Enter your selection :
>1
>
>Create threshold, Update threshold, or Exit (c,u,<e>)?                    A
>u
>
>Current Threshold Definitions
>1) $DERIVED$:Derived:Derived Category:Derived Subcategory:UserCpuUtil:::   B
>2) $DERIVED$:Derived:Derived Category:Derived Subcategory:RespTimeGoal:::
>3) CLO1:IO Activity:::Disk IOs/Sec::::
>4) CLO1:IO Activity:Disk:Summary:Percent Busy:DSKF1:::
>
>Enter number, name, <blank> for more, or E to exit :
>1
>
>Select the resource(s) to use for this derived statistic.                 C
>
```

```
>Resource Name (precede choice with '-' to deselect)
>1) * IP000
>2) * IP001
>3)   IP002
>4)   IP003
>
>Enter number, name, <blank> for more, or E to exit :                    D
>-2
>
>Resource Name (precede choice with '-' to deselect)
>1) * IP000
>2)   IP001
>3)   IP002
>4)   IP003
>
>Enter number, name, <blank> for more, or E to exit :                    E
>E
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A To update a threshold, enter **u** at the prompt.

B The **tqzip** program displays a list of the parameters and derived statistics that currently have thresholds defined. Select the derived statistic threshold definition that you want to update. You can enter either the number or the full name of your selection. Derived statistic names begin with $DERIVED$.

C The derived statistic qualifier lists are displayed and you are given the chance to update your selections. To add a selection to the qualifier list, you enter the number or name of the qualifier value. To remove a selection from the list, you enter the number of the qualifier values preceded by a hyphen (-). An asterisk is used to indicate which qualifier values in the list have been selected. You continue adding or removing selections in the qualifier list until entering an E to exit.

D Note the usage of the hyphen (-) to remove IP001 from the qualifier list.

E After you select a parameter name, the **tqzip** program displays the series of threshold definition prompts with the current value set as the default for each prompt (see G through P in 7.4). You can modify any of the fields.

7.10. Viewing Derived Statistic Threshold Definitions

The **tqzip** program allows you to view an existing derived statistic threshold definition. Select View Thresholds on the Alarms and Thresholds Menu (see 7.3) by entering **2** at the prompt. The **tqzip** program then prompts you with a list of parameter and derived statistic names that currently have threshold definitions in the database as shown in the following example.

```
>Enter your selection :
>2
>
>Current Threshold Definitions                                                A
>1) $DERIVED$:Derived:Derived Category:Derived Subcategory:UserCpuUtil:::
>2) $DERIVED$:Derived:Derived Category:Derived Subcategory:RespTimeGoal:::
>3) CL01:IO Activity:::Disk IOs/Sec::::
>4) CL01:IO Activity:Disk:Summary:Percent Busy:DSKF1:::
>
>Enter number, name, <blank> for more, or E to exit :
>1
>
>
>$DERIVED$:Derived:Derived Category:Derived Subcategory:UserCpuUtil:::    B
> 1.  Threshold checking : ON
> 2.  Threshold 1        : 00075.00  Threshold 2     : 00000.00
> 3.  Number of zones    : 2         Normal zone     : 1
> 4.  Secs before alarm  : 0000      Secs delay      : 0000
> 5.  Alarm identifier   : UserCpuUtil
> 6.  User text :  Call John at ext 2332
> 7.  User text :
> 8.  User text :
>
>List System Names (<y>,n)?                                                   C
>y
>
>   venus
>
>List Resource Names (<y>,n)?
>y
>
>   IP000
>   IP001
>
>Do you want to view another threshold (y,<n>)?                               D
>N
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A The **tqzip** program displays the list of threshold definitions currently defined in the performance database.

Select the definition you want to view. You can enter either the number of the parameter or the full name at the prompt. Derived statistic threshold names begin with $DERIVED$.

B The selected threshold definition is displayed.

C Enter **y** at the prompt to view each qualifier list for this derived statistic threshold. The qualifier lists for system names and resource names are shown in the example.

D If you enter **y** at the final prompt, the **tqzip** program prompts you for another parameter name to view.

If you enter **n** at the prompt, the **tqzip** program returns to the Alarms and Thresholds Menu (see 7.3).

7.11. Deleting Thresholds

The **tqzip** program allows you to delete an existing threshold definition. Select Delete Threshold on the Alarms and Thresholds Menu (see 7.3) by entering **3** at the prompt. The **tqzip** program then prompts you with a list of parameter and derived statistic names that currently have threshold definitions in the database as shown in the following example.

```
>Enter your selection :
>3
>
>Current Threshold Definitions
>1) $DERIVED$:Derived:Derived Category:Derived Subcategory:UserCpuUtil:::   A
>2) $DERIVED$:Derived:Derived Category:Derived Subcategory:RespTimeGoal:::
>3) CL01:IO Activity:::Disk IOs/Sec::::
>4) CL01:IO Activity:Disk:Summary:Percent Busy:DSKF1:::
>
>Enter number, name, <blank> for more, or E to exit :
>1
>
>$DERIVED$:Derived:Derived Category:Derived Subcategory:UserCpuUtil:::      B
>Delete this threshold (y,<n>)?
>y
>
>The threshold definition has been deleted.
>
>Do you want to delete another threshold (y,<n>)?                            C
>
>
>Threshold checking is ON.
>Notify active probes of these threshold changes (y,<n>)?                    D
>y
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A The **tqzip** program displays the list of threshold definitions currently defined in the performance database. Select the one you want to delete by either entering the number or full name at the prompt. Derived statistic names begin with $DERIVED$.

B The **tqzip** program displays the name of the selected parameter or derived statistic for verification before the deletion is performed. If you enter **y** at the prompt, the threshold definition is deleted from the performance database. If you enter **n** at the prompt, the threshold definition is not deleted from the performance database.

C If you want to delete another threshold definition, enter **y** at the prompt. The **tqzip** program returns you to the prompt that allows you to select another threshold for deletion.

D When any changes have been made to the threshold definitions in the performance database, the **tqzip** program displays a final prompt. If overall threshold checking is turned on, you must notify any active probes of the threshold definition changes before they will take effect. Enter **y** at the prompt to notify the probes. You can also use the Notify Probes of Threshold Changes selection on the Alarms and Thresholds Menu to perform this task (see 7.12).

If overall threshold checking is turned off, the **tqzip** program displays the following prompt:

```
>Threshold checking is OFF.
>Do you want to turn it ON (y,<n>)?
>y
>
>Threshold checking is now ON.
>*********************************************
>*   TQALM must be executing on your system     *
>*       when threshold checking is ON.         *
>*********************************************
```

Overall threshold checking must be turned on before any of the individual threshold definitions are checked for threshold violations. To turn on overall threshold checking, you can either answer **y** to this prompt or use the Turn Threshold Checking On/Off selection on the Alarms and Threshold Menu to perform this task (see 7.3).

When using threshold checking, you must also ensure that the **tqalm** program is executing on your system. See 7.17 for information on the **tqalm** program and alarm handling.

7.12. Notifying Probes of Threshold Changes

The TeamQuest Baseline probes use the threshold definitions created by **tqzip** to determine the threshold checking to be done. When a probe is initialized, the threshold definitions are read from the database. The probe uses this initial set of threshold definitions during its entire execution unless it is notified that threshold changes have been made.

To notify all active TeamQuest Baseline probes that changes have been made to threshold definitions, use the Notify Probes of Threshold Changes selection on the Alarms and Thresholds Menu (see 7.3). Enter **4** at the prompt and press the transmit key. Upon being notified, the probes re-read the threshold definitions from the database and obtain the latest set of thresholds available.

You must notify the probes of any threshold changes after defining, updating, or deleting threshold definitions. If you do not notify the probes of the threshold changes, the changes do not take effect until the probes are reinitialized.

7.13. Unloading Threshold Definitions

The **tqzip** program allows you to export all the threshold definitions currently contained in the performance database into a specified file. You can then use this file to view all threshold definitions, update the threshold definitions using a text editor, and reload them at a later time by using the -l and -t options on the **tqzip** processor call (see 7.14).

To unload threshold definitions with **tqzip**, use the following processor call format:

```
@tqzip [-d,database,] -u,-t,threshold-file
```

where:

-d specifies the database that **tqzip** uses. The *database* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-u specifies that an unload operation is to be performed by **tqzip**. The type of unload operation is “thresholds” as indicated by the -t option.

-t unloads the threshold definitions from the database to the file specified by the *threshold-file* parameter.

Example

The following shows an example of unloading threshold definitions to the file tfile.

```
>@tqzip -u,-t,tfile
>TQZIP 7R4G (011312 11:04:45) Fri Jan 13 15:37:08 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>
>Using database TQDATA:PRODUCTION
>
>...unloading thresholds to: tfile
>
>4 threshold definitions were unloaded.
>
>End TQZIP
```

Threshold Definition File Format

The threshold definitions appear in the following format in the output file created by **tqzip**. If you update the threshold definition file, you should ensure that the format of the file remains consistent with what was output by the -u and -t options. The following example shows how the definitions appear in the file:

```
BEGIN THRESHOLDS unloaded on Fri Jan 13  10:15:49 2012                  A
$DERIVED$:Derived:Derived Category:Derived Subcategory:UserCpuUtil:::   B
 1. Threshold checking  : ON                                            C
 2. Threshold 1         : 00075.00  Threshold 2   : 00000.00
 3. Number of zones     : 2         Normal zone   : 1
 4. Secs before alarm   : 0000      Secs delay    : 0000
 5. Alarm identifier    : UserCpuUtil
 6. User text : Call John at extension 2332
 7. User text :
 8. User text :
 9. System Name Count : 1                                               D
    CL01
10. Resource Name Count : 2
    IP000
    IP001
11. Workload Name Count : 0
12. Application Group Name Count: 0
CL01:IO Activity:Disk:Summary:Percent Busy:DSKF1:::                     E
 1. Threshold checking  : ON                                            F
 2. Threshold 1         : 00050.00  Threshold 2   : 00075.00
 3. Number of zones     : 3         Normal zone   : 1
 4. Secs before alarm   : 0060      Secs delay    : 0060
 5. Alarm identifier    : Percent Busy
 6. User text : When more than 50% busy, call John at extension 2332.
 7. User text :
 8. User text :
 END THRESHOLDS                                                         G
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A The unload output file starts with a BEGIN THRESHOLDS line. This line must not be changed or deleted.

B This is the first line of a threshold definition for a derived statistic. Each derived statistic threshold definition consists of a derived statistic name line followed by eight lines of descriptive information and four lines for qualifier values. The derived statistic name line must specify a valid derived statistic name.

C The descriptive lines must be in order from 1 through 8. All text in these lines must remain intact.

You can update the field values (that is, the values following the colons) as long as you keep the fields in their same relative position on each line. For example, if you are modifying the descriptive information of line 3, do not alter the text “Number of zones” or “Normal zone”. You can, however, change the field value for the number of zones from 3 to 2. The spaces surrounding field values are insignificant.

D Lines 9 through 12 are for the qualifier values selected for this threshold definition. The numbered line contains the qualifier name and number of values that follow. The values are each on a separate line.

You can add additional threshold definitions by duplicating all the lines of a threshold definition and modifying the derived statistic name line and the descriptive information lines for the new threshold. Remember to include the qualifiers needed to calculate the derived statistic values for threshold checking.

E This is the first line of a threshold definition for a parameter. Each parameter threshold definition consists of a parameter name line followed by eight lines of descriptive information. The parameter name line must specify a valid parameter name.

F The descriptive lines must be in order from 1 through 8. All text in these lines must remain intact.

You can update the field values (that is, the values following the colons) as long as you keep the fields in their same relative position on each line. For example, if you are modifying the descriptive information of line 3, do not alter the text “Number of zones” or “Normal zone.” You can, however, change the field value for the number of zones from 3 to 2. The spaces surrounding field values are insignificant.

You can add additional threshold definitions by duplicating these lines and modifying the parameter name line and the descriptive information for the new threshold.

G The last line of the thresholds file must be the END THRESHOLDS line. You must not change or delete this line.

7.14. Loading Threshold Definitions

You can reload the performance database with a new or updated set of threshold definitions by using the -l and -t options on the **tqzip** processor call. You should use a file as created by the -u and -t options (see 7.13) as input for loading threshold definitions. If you update the threshold definition file, you should ensure that the format of the file remains consistent with what was output by the -u and -t options.

Notes:

- *The **tqzip** program deletes any existing threshold definitions from your database before loading the new definitions. Use the -e option on **tqzip** if you do not want existing definitions deleted.*
- *After the threshold definitions have been loaded into the database, all active TeamQuest Baseline probes are automatically notified of the threshold changes. They begin using the new threshold information with the next data sample they collect.*

To load threshold definitions with **tqzip,** use the following processor call format:

```
@tqzip [-d,database,] -l,-t,threshold-file [-e]
```

where:

-d specifies the database that **tqzip** uses. The *database* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-l specifies that a load operation is to be performed by **tqzip**. The type of load operation is “thresholds” as indicated by the -t option.

-t loads the threshold definitions into the database from the files specified by the *threshold-file* parameter. The *threshold-file* parameter is the name of a threshold definition file created by the **tqzip** -u and -t options (see 7.13).

-e loads the new threshold definitions from the threshold definition file without first deleting the existing definitions in the database. If you do not specify the -e option, all existing definitions are deleted from your database before the new ones are loaded.

Example

By using the -l and -t options of the **tqzip** processor call, you can use different sets of threshold definitions at various times of the day.

For example, you may want to use different threshold values for your day shift than you use for the night shift. You can do this by performing the following steps:

1. Using **tqzip**, define the thresholds you want to use for your day shift (see 7.4).
2. Unload the current set of threshold definitions to a file called DAYTHRSHLDS (see 7.13).
3. Use FURPUR to copy the DAYTHRSHLDS to a new file called NITETHRSHLDS.

4. Edit NITETHRSHLDS to contain the threshold values you want to use during the night shift.
5. Schedule two separate batch jobs to be run each day. The first job will run at the start of the day shift. It will execute **tqzip** with the -l and -t options to load the threshold definitions from the DAYTHRSHLDS file. The second job will run at the start of the night shift. It will load the threshold definitions from the NITETHRSHLDS file.

7.15. Turning Threshold Checking On and Off

The **tqzip** program allows you to control whether or not the TeamQuest Baseline probes perform threshold checking on the data they collect. When threshold checking is turned off, the probes do not check the parameter values for threshold conditions. When threshold checking is turned on, the parameter values collected by the probes are checked at each sample for threshold violations.

7.15.1. Using the Alarms and Thresholds Menu

To turn off or on threshold checking, select Turn Threshold Checking On/Off from the Alarms and Thresholds Menu (see 7.3) by entering **5** at the prompt.

If threshold checking is currently turned on for the performance database, the **tqzip** program displays the following prompt to turn off threshold checking:

```
>Enter your selection :
>5
>
>Threshold checking is ON for TQDATA:PRODUCTION
>Enter Y to reverse this setting :
>y
>
>Threshold checking is now OFF.
```

All active TeamQuest Baseline probes automatically stop threshold checking at the next data sample when Threshold Checking is changed from ON to OFF.

If threshold checking is currently turned off for the performance database, **tqzip** displays the following prompt to turn on threshold checking:

```
>Enter your selection :
>5
>
>Threshold checking is OFF for TQDATA:PRODUCTION
>Enter Y to reverse this setting :
>y
>
>Threshold checking is now ON.
>
>*****************************************
>* TQALM must be executing on your system  *
>*      when threshold checking is ON.     *
>*****************************************
```

All active TeamQuest probes automatically begin threshold checking at the next data sample when Threshold Checking is changed from OFF to ON.

7.15.2. Using the tqzip Processor Call

To turn off or on threshold checking or to view the current setting for threshold checking, use the following processor call format:

```
@tqzip [-d,database,] -a,action
```

where:

-d specifies the database that **tqzip** uses. The *database* parameter is the name of an existing database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-a indicates the action to be taken with the threshold checking setting. *action* can be one of the following:

ON	turns threshold checking on
OFF	turns threshold checking off
LIST	displays the current threshold checking setting

7.16. Generating Alarm Lists

You can look at the alarm messages that have been generated by the TeamQuest Baseline probes by using the Generate Alarm List selection on the Alarms and Thresholds Menu (see 7.3). The **tqzip** program generates a list of the alarm messages and places them in a specified file. In order to generate alarm lists through **tqzip**, the **tqalm** program must be executing on your system. (See 7.17 for information on how to start **tqalm**.)

To view the alarm messages with **tqzip**, select Generate Alarm List from the Alarms and Thresholds Menu (see 7.3) by entering **6** at the prompt. The following example shows the prompts displayed when you select Generate Alarm List from the Alarms and Thresholds Menu:

```
>Enter your selection :6
>6
>
>Enter the output filename :                                          A
>alrmfile
>Enter start date & time yy/mm/dd hh:mm:ss :                          B
>
>Enter end date & time yy/mm/dd hh:mm:ss :
>
>
>...outputting alarm messages to: alrmfile                            C
>
>2 alarm messages were selected.
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A The **tqzip** program prompts you for the file name for your alarm messages file.

B The **tqzip** program also prompts you for a starting date and time, as well as an ending date and time. You can limit the number of messages that are output to the alarm messages file by specifying a starting and ending date and a starting and ending time.

If you do not enter a starting date or time, **tqzip** uses the first date and time found in the alarm file. If you do not enter an ending time, all messages after the starting time are output.

C The **tqzip** program generates the list of alarm messages and places them in the alarm messages file you specified as the output file name. (See 7.18 for a description of the format of alarm messages.)

Alarm Messages File Format

The following shows how the alarm messages appear in the output file:

```
>All available alarm messages follow:
>
>Fri Jan 13 08:25:33 2012
>*TQ* CRITICAL: <12.51> CL01 Demand Response Time
>      Call John at extension 2332
>
>Mon Dec 19 15:29:02 2011
>*TQ* CRITICAL: <43212.00>  CL01 Fixed Tracks Available
>
>End of alarms.
```

7.17. Using tqalm

The TeamQuest Baseline probes perform threshold checking for all parameters that have a threshold definition in the performance database. When an alarm condition is encountered, the probe generates an alarm. Alarms are handled by the **tqalm** program that is provided with TeamQuest Baseline. The **tqalm** program can perform the following tasks for each alarm:

- Send an alarm message to the system console
- Call a user-provided routine for site-specific alarm handling
- Optionally, send an SNMP trap to a network management console (see Appendix J)

In addition to alarm handling, the **tqalm** program performs threshold checking for all derived statistics that have a threshold definition in the performance database. Every *n* seconds (where *n* is the granularity of the aggregation set used by **tqalm**), **tqalm** retrieves the data needed to calculate derived statistic values, performs threshold checking on the values, and generates alarms as needed. To allow **tqalm** to do threshold checking using the most current data available, it is important that it uses the same database and finest granularity aggregation set that is being used by the data collection probes running continuously on your system (that is, **tqbsp**). Use the -d and -a options the **tqalm** processor call to specify the database and aggregation set.

The **tqalm** program must be running on your system in order for any alarm handling to occur. To start **tqalm**, use the **tqalm** runstream in SYSLIB*RUN$. The **tqalm** runstream is started automatically when the TeamQuest Baseline control runstream **tqctl** is started. See 9.2 for more information about **tqctl**.

Processor Call Format

The processor call format for **tqalm** is

```
@tqalm [-a,aggset,] [-d,database,] [-c,] [-w,delaysecs,]
       [-m,size,] [-r]
```

where:

-a specifies the aggregation set used by **tqalm** to obtain the data values needed to calculate the values for derived statistic threshold checking. If the -a option is not specified, the AGGSET1 setting in the tq-ini element (see 3.1) is used. The granularity of the aggregation set is used to determine how often to check the derived statistic thresholds.

-d specifies the database that **tqalm** uses. The *database* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1). The **tqalm** program uses this database to retrieve information for the derived statistic thresholds it needs to be checking.

-c sends a console message for each alarm that is generated.

-w specifies the delay factor in seconds to use in derived statistic threshold checking. By default, **tqalm** uses the most recent data point stored to calculate derived statistic values for threshold checking. Using the *delaysecs* parameter gives all the TeamQuest probes time to write their data values before the derived statistic values are calculated for threshold checking.

The number of seconds specified as *delaysecs* is rounded up to the nearest whole number of samples by using the aggregation set granularity. For example, if granularity is 60 seconds and you specify a delay of 1 to 60 seconds, the **tqalm** program will delay one sample. That means that **tqalm** will not use the most recent data point, but the point before the most recent one to do derived statistic threshold checking.

-m is used with the -r option for real-time mode execution of **tqalm**. The *size* parameter is used to allocate memory space before going to real-time mode in **tqalm**. The *size* parameter is specified in thousands of words and the default is 65 (that is, 65,000 words).

-r is used to execute **tqalm** in real-time mode. A real-time priority of 35 is used to run **tqalm**.

If you have problems running in real-time mode, you can use the -m option to adjust the size of memory space allocated by **tqalm** before it goes to real-time mode.

Terminating the tqalm Run

To terminate the **tqalm** run from a demand run that has full console mode capabilities or greater, enter

```
@@CONS TQALM END
```

To terminate the **tqalm** run from the system console, enter

```
TQALM END
```

7.18. Alarm Messages

The **tqalm** program generates alarm messages for each alarm that occurs on your system. The alarm messages can be sent to the system console and are written to the TeamQuest Baseline alarm history table by **tqalm**. The alarm messages displayed within a TeamQuest View session and those viewed within **tqzip** (see 7.16) are retrieved from the alarm history table.

Alarm messages can consist of up to four lines of text. The first line is generated by TeamQuest Baseline. The next three lines of text are the user alarm message text lines that you specified when creating the threshold definition for the parameter (see 7.4). The alarm message for a parameter has the following format:

```
*TQ*  severity: <xxxxxx.xx> system alarm-id qual1 qual2 qual3
       user alarm message line 1
       user alarm message line 2
       user alarm message line 3
```

where:

severity	is NORMAL, WARNING, or CRITICAL. WARNING and CRITICAL indicate the raising of an alarm. NORMAL indicates the clearing of an alarm. See 2.6 for more information on alarm severities.
<xxxxxx.xx>	is the value of the parameter when the alarm occurred.
system	is the system name on which the alarm occurred.
alarm-id	is the alarm identifier specified in the threshold definition of the parameter.
qual1	is the resource and/or application group name for the parameter.
qual2	is the workload set for the parameter.

qual3	is the workload for the parameter.
user alarm message line 1, 2, 3	is the user-defined text as specified in the threshold definition for the parameter. The alarm message can contain up to three lines of user alarm message text. These lines are optional and are not displayed with the alarm when the severity is NORMAL.

Example

The following alarm message is created by **tqalm** if a threshold violation has occurred on your system for fixed tracks available:

```
*TQ* CRITICAL: <43212.000> CL01 Fixed Trks Available
```

CRITICAL indicates the alarm severity and 43212.000 is the current value for the parameter. CL01 is the name of the system on which the alarm occurred. Fixed Trks Available is the alarm identifier that was specified when the threshold definition was created using **tqzip**. In this example, *qual1*, *qual2*, and *qual3* do not exist for this parameter.

7.19. Site-Specific Alarm Handling

You can perform site-specific actions to handle alarms by writing a C language function that is called by the **tqalm** program. For each alarm that is generated, **tqalm** calls a function named usralarm. It passes the address of a data structure to this function that contains information on the alarm that has just been generated. For a description of this data structure, you can look at the SYSLIB*TQ-BASELINE.USRALARM-C element.

By default, the usralarm function returns to **tqalm** without performing any action on the alarm. To add site-specific code, you must update SYSLIB*TQ-BASELINE.USRALARM-C and compile it using the UC compiler. You then need to copy the USRALARM-C object module created by the UC compiler to TPF$ and enter the following control statement to relink **tqalm**:

```
@LINK SYS$LIB$*TQ-BASELINE.LNK-USRALARM, SYS$LIB$*TQ-BASELINE.TQALM
```

You must have write access to SYSLIB*TQ-BASELINE to put the new **tqalm** executable back into that file.

7.20. Single Point Operations (SPO) Considerations

Note: *This subsection applies only if you use the Unisys Single Point Operations (SPO) products.*

The SPO products monitor and control multiple computer systems from one operations console. The message automation and alarm handling functionality of SMART or the Single Point Autoaction Message System (SP-AMS) and the alarm capability of TeamQuest Baseline can be integrated by following the guidelines listed in this subsection.

The two key components of integrating TeamQuest Baseline and SPO are the TeamQuest Baseline threshold definition and the associated SMART/SP-AMS pattern that is used to process the console alarm message generated when an alarm is raised or cleared by TeamQuest Baseline. SMART or SP-AMS transforms the TeamQuest Baseline message into an event report that tells SPO and Status the necessary information to create displays and carry out automated alarm actions. In addition, SMART or SP-AMS may directly take automated actions based on the message.

The alarm identifier and the user text lines specified when defining thresholds with **tqzip** (see lines `N` and `O` in 7.4) are used to build alarm messages that are sent to the system console by TeamQuest Baseline. Certain conventions should be followed when defining thresholds using **tqzip** to ensure easy pattern matching and data extraction using SMART/SP-AMS.

- Alarm Identifier

  Each threshold should have a unique alarm identifier with no embedded spaces. The unique alarm identifier reduces the possibility of an unintended pattern match. By default, SMART/SP-AMS breaks messages into tokens using spaces as delimiters. By ensuring that the alarm identifier does not contain spaces, it makes extracting the identifier token easy and adds some measure of predictability to the message.

- User Text Lines

  User-defined alarm text can be defined as three separate messages using **tqzip**. Each of these messages is displayed on a separate line in the TeamQuest Baseline alarm message. This causes multiple output lines on the console. Since SMART/SP-AMS requires you to use variables to correlate multiple-line console messages, you can simplify the pattern definition and avoid the use of variables by defining any user text for the alarm within SPO itself.

  If you define multiline console messages with **tqzip**, then putting the unique alarm identifier on each line, along with a sequence number, will reduce the possibility of an unintended pattern match.

  The user text lines within TeamQuest Baseline can still be used if you want the user text to appear in the alarm message generated by TeamQuest Baseline. It is important to remember, however, that because this user text is not on the first line of the alarm message it will not be available within the SMART/SP-AMS pattern matching done for the alarm.

The alarm identifier in each TeamQuest Baseline alarm message should be used within SMART/SP-AMS pattern matching to determine which alert needs to be raised or cleared. This alarm identifier can be used as the alarm identifier in the alarm event report SMART/SP-AMS creates and sends to SPO. If you include the appropriate user text in the alert event report, it will be displayed by SPO in the Alerts and Alert Detail windows. In addition, you can define supplemental help text for the alarm and keep this file on the SPO server. It will be displayed in the Alert Detail window.

SPO lets you define alarm policies that include taking actions when alarms are raised. These actions include dialing a pager, both immediately and after a timed delay. This permits you to notify a site support person when TeamQuest Baseline discovers a performance anomaly. Then, if the problem has not been corrected after a certain amount of time, you can automatically escalate the notification.

The TeamQuest Baseline alarm message contains an indicator of the alarm severity. The indicators are NORMAL, WARNING, and CRITICAL. The alarm severities can be used within SMART/SP-AMS pattern matching to determine when to raise or clear an alert. A severity of WARNING or CRITICAL indicates that an alert should be raised. A severity of NORMAL indicates that the alert can be cleared.

Single Point Status provides enhanced monitoring of systems and their components as an add-on to SPO. For TeamQuest Baseline alarm messages to drive status displays, the message and pattern matching considerations described previously apply. Single Point Status uses Attribute Change (AC) and Delete (DE) event reports rather than Alarm (AL) event reports, but it provides the capability to automatically create alarm event reports when a monitored attribute attains a user-specified value or enters a user-specified range.

In contrast with SPO, Status diagrams show not only normal (green) and alarm (red) states, but also an attention state (yellow) that can indicate a component whose performance is worse than normal but not yet of alarm significance. If you use Status to show the SPO operator critical data reported by TeamQuest Baseline, then you will probably find it best to use SMART/SP-AMS to create AC and DE event reports and let Status automatically generate the AL event reports.

See the SPO documentation for details of patterns, pattern matching and actions, event reports, alarm help text, and external alarm actions.

For more information on SMART/SP-AMS, see the *Single Point Autoaction Message System Administration Guide* and the *SMART for ClearPath OS 2200 Administration and Operations Guide*.

Other Considerations

You will need to define TeamQuest Baseline thresholds locally on each monitored system. No centralized definition point exists. However, you may be able to use the threshold load/unload capability in **tqzip** to assist in copying threshold definitions from one system to another (see 7.13 and 7.14).

Section 8
Customizing Derived Statistics

8.1. Introduction

TeamQuest Performance Software Administration or the **tqzip** program can be used to administer the derived statistics in your performance database. If you want to customize your derived statistics, you can perform the following tasks:

- Create derived statistics
- Delete derived statistics
- Modify derived statistics
- View derived statistics

8.2. Concepts

Some basic concepts are essential to the understanding of TeamQuest Baseline derived statistics. You should have an understanding of these concepts before beginning your administrative tasks with derived statistics.

8.2.1. Derived Statistic Identification

A derived statistic is identified in the database using the TeamQuest Baseline key name hierarchy as described in 2.2.2. Derived statistic definitions apply to all systems in your database. Because of this, a derived statistic is actually identified by using only the category group, category, subcategory, and statistic key names.

When you create a derived statistic with TeamQuest View, you are allowed to insert it at the statistic level at any point in the existing parameter hierarchy. A special location exists in the hierarchy if you want to group all of your derived statistics together. This special location is identified by:

Category Group	“Derived”
Category	“Derived Category”
Subcategory	“Derived Subcategory”
Statistic	<your statistic name>

8.2.2. Parameter References and Substitution

Derived statistic formulas can reference the values of other derived statistics or of parameters being collected by the TeamQuest Baseline data collection probes. This allows you to calculate new statistics based on existing ones. Parameter references are made by specifying all of the key names used to identify a parameter in the database. (See 2.2.2 for a description of the parameter key name hierarchy.)

To keep the formula definition itself as short and concise as possible, you define short names, called aliases, for each of the parameters you wish to reference from within the formula. The list of aliases used in your formula is specified when you create your derived statistic using TeamQuest Performance Software Administration.

Parameter references in derived statistic formulas can be either specific references that contain the entire set of key names needed to identify the parameter or they can contain substitution indicators. Substitution indicators are used to tell TeamQuest Baseline to fill in the values for selected keys at report time. Substitution is allowed for system, workload set, workload, resource, and application group. The following special strings are used in a parameter reference to indicate that substitution is to take place:

Individual Substitution	**Group Substitution**
$SYSTEM	$ALL_SYSTEM
$WLS	$ALL_WLS
$WL	$ALL_WL
$RES	$ALL_RES
$AG	$ALL_AG

When using individual substitution, the parameter reference is expanded into a set of individual parameter references using each of the values provided for the specified key. The values used for substitution are obtained from either the report definition file or from prompting that is done when the report is opened. The solving of the single derived statistic formula will actually report the values for the set of individual parameters. For example, if $RES is used in a parameter reference for a disk statistic, the formula is solved once for each disk specified for the report. If four disks were selected when the report was opened, the values for each of these four disks would be reported as a result of using this single derived statistic.

When using group substitution, all the values for the specified key are used to expand out the list of arguments to a mathematical function. Group substitution is only allowed in the argument list of a mathematical function. The argument list is expanded as though a separate parameter reference was used specifying each key explicitly. For example, to find the average %busy of all your disks, you would use the AVG function and include a single parameter reference in its argument list. The single parameter reference would contain $ALL_RES for the resource key specification. This would expand into one reference for each disk on your system. The AVG function would average all of these values and return a single value for the derived statistic. Unlike individual substitution, only one value is reported as a result of resolving a formula using group substitution.

In a single formula, all parameter references that use substitution must be of the same type. For example, you cannot use $RES in two parameter references unless they are both for the same type of resource (for example, either tape or disk). This means that using both tape and disk substitution in one formula is not allowed. Parameters using group substitution have no associated type and their usage is not limited as with individual substitution.

8.2.3. Formula Definition

A derived statistic formula is used to calculate the value that is reported for the statistic. This formula is entered when creating a derived statistic definition using TeamQuest Performance Software Administration.

Formulas can consist of parameter references (by using aliases), arithmetic operators, constants, and mathematical functions. The MIN, MAX, AVG, SUM, and GRAN functions are supported. The MIN, MAX, AVG, and SUM functions require a list of values as input. The value list is made up of one or more values separated by commas. A value can be either a numeric constant or a parameter reference alias. TeamQuest Baseline obtains the value for each parameter reference and inserts it into the function call.

MIN (*value*, ...)
: This function compares all the values and returns the lowest of the values.

MAX (*value*, ...)
: This function compares all the values and returns the highest of the values.

AVG (*value*, ...)
: This function calculates and returns a simple average using the values passed as input.

SUM (*value*, ...)
: This function returns the sum of all the values passed as input.

Derived parameter values are calculated at report time. This calculation is done using the aggregation set specified for the report. The GRAN function can be used to obtain the granularity of the aggregation set being used in the report. This function is called without any parameters:

GRAN ( )
: This function returns the granularity (in seconds) of the aggregation set being used for the current report.

8.3. Maintaining Derived Statistics with TeamQuest Performance Software Administration

To perform any administration tasks using TeamQuest Performance Software Administration, you must have a session established to the host software (see 4.2). You access the dialog boxes for performing administration tasks by selecting the Administration menu.

Note: *When derived statistics maintenance tasks are being performed, the database is locked. Only the user performing the tasks can access the database for modifying or updating derived statistics. Other users can view derived statistic information in read-only mode when the database is locked.*

TeamQuest Performance Software Administration allows you to perform the following tasks:

- Performing Derived Statistic Administration Tasks (see 8.3.1)
- Creating a New Derived Statistic Definition (see 8.3.3)
- Saving a Derived Statistic Definition (see 8.3.4)
- Viewing Long Key Names (see 8.3.5)
- Importing Derived Statistic Definitions (see 8.3.6)
- Exporting Derived Statistic Definitions (see 8.3.7)
- Displaying or Modifying a Derived Statistic Definition (see 8.3.8)
- Deleting a Derived Statistic Definition (see 8.3.9)

Note: *It is recommended that you use the Administration utility to create and update your derived statistic definitions.*

8.3.1. Performing Derived Statistic Administration Tasks

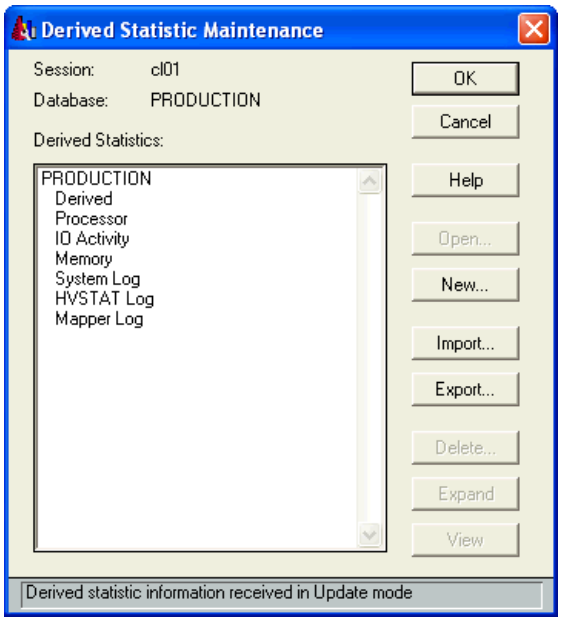
To begin any derived statistic maintenance task, select Administration > Derived Stats.

If more than one session is open, the Administration Session Selection dialog box is displayed. This dialog box contains a list of open sessions for which you can perform derived statistic maintenance and session information for the session highlighted in the list box. Select the desired session and click OK.

If more than one database is available for the session, the Administration Database Selection dialog box is displayed. Select the database for which you want to perform derived statistic maintenance and click OK.

The Derived Statistic Maintenance dialog box is displayed.

Note: *As the derived statistic information is loading, the status bar at the bottom of the dialog box indicates the percentage of information loaded. When the percentage reaches 100, the message disappears and you are ready to perform derived statistic maintenance tasks.*



All derived statistic maintenance tasks can be performed through this dialog box. The Derived Statistics list box contains a hierarchical list of the derived statistics that are defined in the database (see 8.3.2). When the list initially displays, the hierarchy is expanded to the first level that contains more than one derived statistic. The database name is at the top, followed by the category group, category, subcategory, and statistic.

The status bar at the bottom of this dialog box displays information about the selected item in the Derived Statistics list box. It displays status information about the derived statistics maintenance tasks or tells you the type of item—database name, category, category group, subcategory, or statistic—that is selected from the list.

All operations are performed relative to a selected key name in the Derived Statistics list box. The Open button opens a selected statistic key to display or modify the derived statistic definition. The New button allows you to create a new derived statistic definition. The Delete button allows you to delete a derived statistic from the Derived Statistics list. The Expand button allows you to expand the statistic hierarchy to the next key level. The View button allows you to look at long key names that may have been truncated in a list box. You must select a statistic key before performing any operations with the Open and Delete buttons.

When you click OK, your changes are made to derived statistic definitions, including any changes made to derived statistics attributes accessed through the New or Open buttons (see 8.3.3 and 8.3.8).

When you click Cancel, any changes you have made are ignored.

8.3.2. Derived Statistic Hierarchy

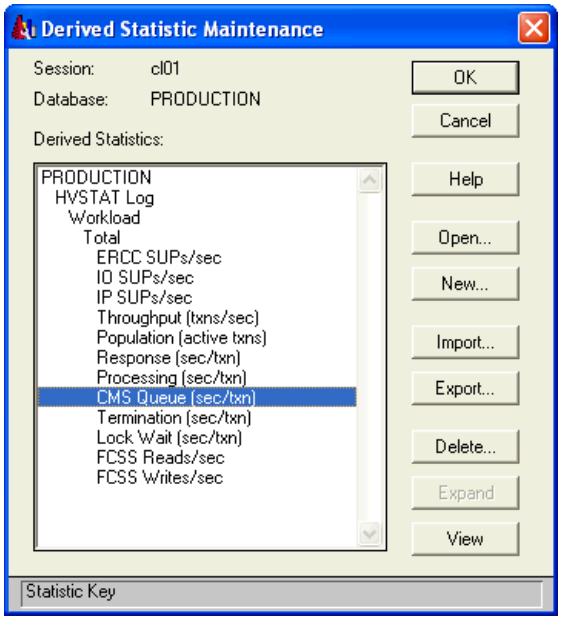
Statistics are stored in the database in a hierarchical format. A statistic is defined by a hierarchy of key names, which identify the statistic for retrieval from the database. See 8.2.1 for a description of this hierarchy. Each layer in the hierarchy is maintained by a key name. The first layer is the first key, the next layer is the next key, and so on. Derived statistic maintenance provides four layers of keys: the category group, the category, the subcategory, and the statistic. All four keys must be supplied when defining a derived statistic. If a key is not actually used for a statistic, null is displayed in the Derived Statistics edit box.

Many of the dialog boxes in derived statistic maintenance contain a statistic hierarchy list. When the list is initially displayed, the hierarchy is expanded to the first level that contains more than one item. The first line of a statistic hierarchy list contains the name of the database you are working on. Indented under the database name are the first level keys, the category groups. Indented under the category groups are the second level keys, category. Subsequent indented levels include the subcategory keys and statistic keys. Each key item is placed on its own line in the list box.

To perform many of the derived statistic maintenance tasks, you must expand the derived statistic list to the full hierarchy, which includes the category group, category, subcategory, and statistic. To define a statistic fully, the statistic key must be selected.

To expand a key item to the next level of the hierarchy, select the item in the Derived Statistics list box and click the Expand button. You can also double-click the item in this list box to display all of the items in the next level of the hierarchy.

The following dialog box shows a derived statistics list fully expanded to show all key levels.



In the Derived Statistics list box, PRODUCTION is the database name; HVSTAT Log is the category group key; Workload is the category key; Total is the subcategory key; and CMS Queue (sec/txn) is the statistic key. Because CMS Queue (sec/txn) is selected, the status bar shows that it is the statistic key.

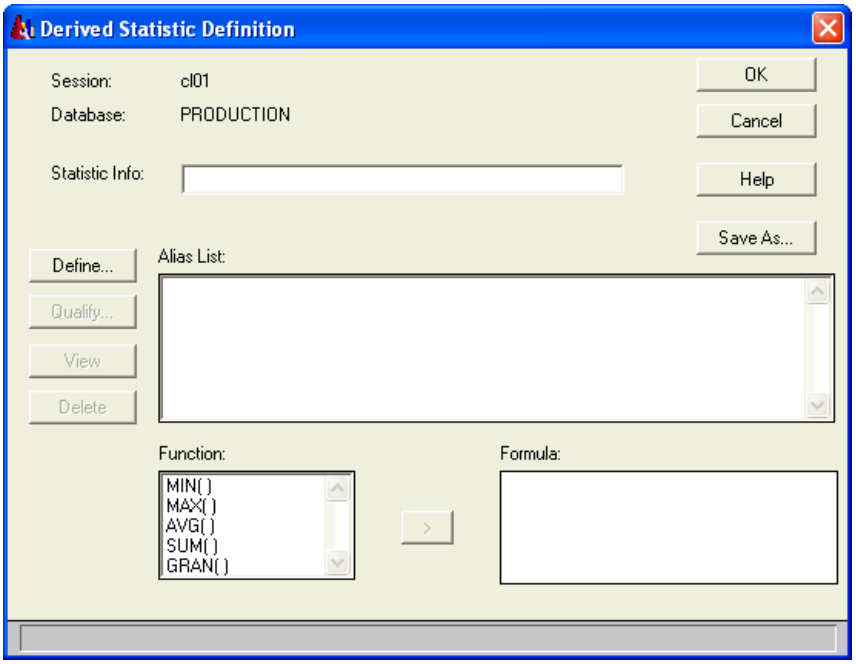
8.3.3. Creating a New Derived Statistic Definition

To create a new derived statistic definition, click the New button in the Derived Statistics Maintenance dialog box (see 8.3.1).

To create a derived statistic definition, do the following:

1. In the Derived Statistic Definition dialog box, enter a description of the derived statistic in the Statistic Info edit box.
2. Click the Define button. Using the statistic hierarchy, access the desired statistic in the database and provide an alias name to be used in the definition of the derived statistic. Do this for each statistic you are using.
3. Use the Formula edit box to develop the derived statistic. Use the alias names in the mathematical expression to develop the derived statistic.
4. Click the Save As button to save the derived statistic in the statistic hierarchy. See 8.3.4 for information on saving a derived statistic.

When you click New, the Derived Statistic Definition dialog box is displayed.



Derived Statistic Definition Dialog Box

The Derived Statistic Definition dialog box contains the following information about the session and database for which you are creating a derived statistic definition:

Statistic Info
: supplies a description of the derived statistic. This information is saved in the database and is available for viewing when subsequently opening a derived statistic definition. This field is limited to 256 characters in length.

Alias List
: provides a list of reference names to other parameters in your database. These reference names can then be used in the formula of a derived statistic. The alias list is maintained by using the Define, Qualify, View, and Delete buttons to the left of the edit box. A separate independent alias exists for each derived statistic definition.

  Define button
  : allows you to add a statistic reference to the alias list (see 8.3.3.1).

  Qualify button
  : allows you to qualify the statistic referenced by the alias name by specifying the complete parameter hierarchy for the statistic. To fully qualify the statistic, you would add the system, resource, workload set, workload, and application group to the derived statistic hierarchy of category group, category, subcategory, and statistic name (see 8.3.3.2).

  View button
  : allows you to view all key name fields for a reference name in the alias list. A key name can be too long to fit in the Alias List edit box and is then truncated. The View button allows you to see the entire key name hierarchy (see 8.3.5).

  Delete button
  : allows you to delete a reference name from the Alias List edit box (see 8.3.3.3).

  When you want to use a reference name in a derived statistic formula, select the item from the Alias List edit box and click the >> button to move the item to the Formula edit box. You can also double-click the item to move it.

Function
: contains a list of server-supported functions that can be used in the formula of a derived statistic. Select the function and click the >> button to move the function to the Formula edit box. You can also double-click the function to move it.

Formula
: provides the mathematical expression that is used to calculate the value for the derived statistic. Formulas can consist of parameter references (by using aliases), arithmetic operators, numeric constants, and mathematical functions.

Aliases are used in a formula to reference parameters being collected by the data collection probes. They can also be used to reference other derived statistics. When an alias refers to a parameter being collected by the data collection probes, the value for the parameter is retrieved from the database and is inserted into the formula at evaluation time. When an alias refers to another derived statistic, the value for the second derived statistic is calculated and the result is inserted into the formula for the first derived statistic at evaluation time. Because you can use aliases to refer to other derived statistics in your formula, be careful that you do not specify a circular definition (that is, a formula that either refers to itself directly or indirectly through the formula of a referenced derived statistic). If this condition is encountered, a warning dialog box informs you of the situation. You must correct the problem before the derived statistic definition can be saved in the database.

In general, the formula is evaluated from left to right. The following operators are supported in order of precedence:

( )	Parenthesis (can be used to provide correct precedence in the formula)
^	Exponentiation
* /	Multiplication and division
+ -	Addition and subtraction

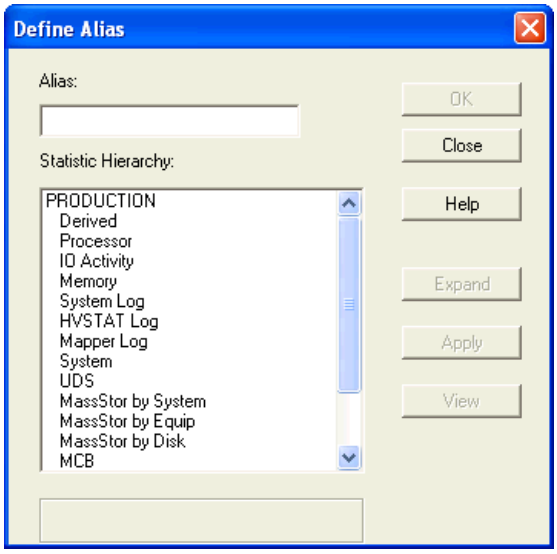
Numeric constants can be specified either with or without a decimal point. However, they are all treated as floating-point numbers. No integer operations are supported. Both positive and negative constants are allowed. Exponential notation is also allowed. For example, 2.0e-3 is the same as .002.

Arguments for mathematical functions are specified as a list of values separated by commas and enclosed in parentheses. Each value can be a numeric constant or a parameter reference alias. You cannot nest mathematical functions by specifying a function name in an argument list.

See 8.2.3 for the mathematical functions supported by TeamQuest Baseline.

8.3.3.1. Defining an Alias

To define an alias, press the Define button next to the Alias List edit box in the Derived Statistics Definition dialog box. The Define Alias dialog box is displayed.



Define Alias Dialog Box

The Define Alias dialog box contains the following fields:

Alias
: provides a short name for the alias. This field is limited to 12 characters in length and must begin with an alpha character from the set A through Z and a through z. The rest of the characters must be alphanumeric.

Statistic Hierarchy
: contains a list of all the statistics defined in the database you are working on, including derived statistics. The statistics are listed in the hierarchy described in 8.3.2. A status box under the list box indicates the key name type (category group, category, subcategory, or statistic) of the item that is selected.

Expand button
: allows you to expand the hierarchy list of a selected key item to the next level.

Apply button
: allows you to send an alias selection back to the Derived Statistic Definition dialog box while staying in the Define Alias dialog box to define more aliases.

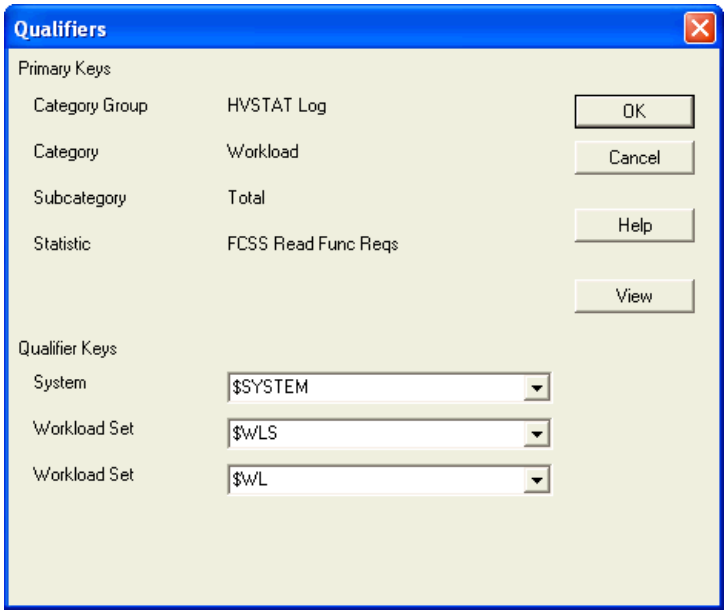
View button
: allows you to see the key values that are truncated in the list box (see 8.3.5).

All of the aliases defined using the Define Alias dialog box refer to unqualified statistics. The statistic itself may have relationships that must be defined to retrieve data for the statistic from the database. Explicitly specifying values for these relationships is called qualifying the statistic. A fully qualified statistic is one whose relationships are all defined such that a value could be retrieved from the database using the given information. For TeamQuest Baseline, all statistics need a system qualifier. In addition, some statistics need to be qualified by resource, or by workload set and workload, or by application group.

The qualification does not need to be explicitly specified because it can be obtained at report time. For all of the aliases defined using this dialog box, required qualifiers are specified using a substitution reference ($SYSTEM, $RES, $WLS, $WL, or $AG). The substitution reference tells the server that these values are filled in at report time. The derived statistic is actually more flexible when substitution is used; however, there may be times when you want to reference fully qualified values. For more information on how to qualify an alias definition, see 8.3.3.2. See 8.2.2 for more information on the qualifier substitution supported by TeamQuest Baseline.

8.3.3.2. Qualifying Parameter References

To qualify the statistic reference for an alias, select the alias from the Alias List edit box of the Derived Statistic Definition dialog box and click the Qualify button. The Qualifiers dialog box is displayed.



The Qualifiers dialog box contains a list of the primary keys—Category Group, Category, Subcategory, and Statistic—for the statistic referenced in the alias. It also contains list boxes for the qualifiers that can be used for the statistic. Qualifier list boxes can appear for system, workload set, workload, resource, and application group. The qualifier list boxes that are displayed depend on the statistic that is being qualified. Only those qualifiers that have a relationship with the statistic are displayed.

The System, Resource, Workload Set, Workload, and Application Group fields contain drop-down edit boxes. A list of all possible values is displayed. If the value you want the qualifier to take on is not provided in the list, you can type in your desired value.

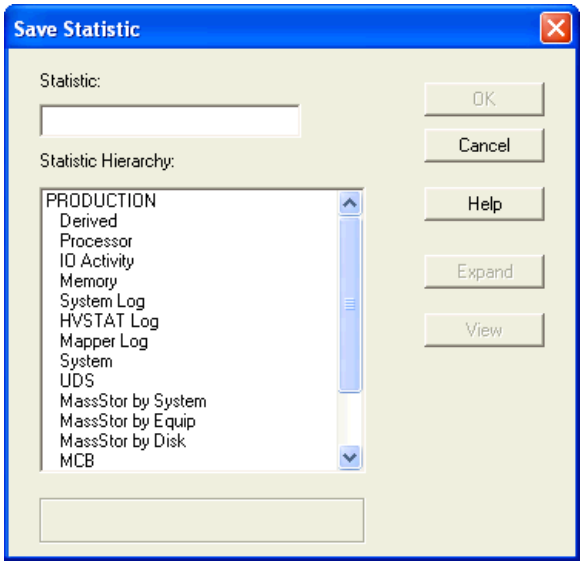
The View button allows you to look at long key names that may have been truncated in the list boxes (see 8.3.5).

8.3.3.3. Deleting an Alias

To delete an alias from the Alias List edit box, select the alias from the Alias list box of the Derived Statistic Definition dialog box and click the Delete button. The alias is removed from the list box. The alias name is not removed from the Formula edit box in the Derived Statistic Definition dialog box. You must remove those references yourself.

8.3.4. Saving a Derived Statistic Definition

To save a newly defined derived statistic, the Save Statistic dialog box is displayed when you click the Save As button from the Derived Statistic Definition dialog box or when you click OK for a new derived statistic definition. This dialog box allows you to specify where to place the new derived statistic in the statistic hierarchy.



Save Statistic Dialog Box

The Save Statistic dialog box contains the following fields:

Statistic
: enter the name for your derived statistic. The statistic name is limited to 51 characters in length and is case-sensitive.

Statistic Hierarchy
: displays the hierarchy for the database you are working on. The status box at the bottom of the list displays the type of key name that is selected in the list. Use the Statistic Hierarchy box to specify where the newly derived statistic is placed in the database hierarchy. Derived statistics can only be placed at the statistic level within the database hierarchy. Therefore, the item selected must be a subcategory or a statistic.

  You must have entered at least one character in the Statistic edit box and have selected a subcategory or statistic item in the Statistics Hierarchy list box before the OK button is enabled. These two items together determine where the newly derived statistic is placed in the statistic hierarchy.

  When you click OK, the new derived statistic is placed in the statistic hierarchy. You must click OK in the Derived Statistics Maintenance window for the newly derived statistic definition to be sent to the database.

  Expand button
  : allows you to expand the hierarchy list of a selected key item to the next level.

  View button
  : allows you to see the key values that are truncated in the list box (see 8.3.5).

8.3.5. Viewing Long Key Names

Many of the dialog boxes contain a View button, which allows you to view complete key name fields that have may be truncated in the dialog box. When you click the View button, the View Statistic Fields dialog box is displayed. This dialog box contains a complete list of all the key names that make up a statistic.

8.3.6. Importing Derived Statistic Definitions

To import all derived statistic definitions, click the Import button on the Derived Statistic Maintenance dialog box (see 8.3.1). When the Open File dialog box is displayed, select the file that contains the derived statistic definitions that you want to import.

When the derived statistic definitions are imported, all existing derived statistic definitions are updated to match the imported definition. New derived statistic definitions are inserted into the Derived Statistic Maintenance dialog box.

If a syntax error occurs, the derived statistic definitions are not inserted in the Derived Statistic Maintenance dialog box.

After an import function, your updates can be deleted by clicking the Cancel button on the Derived Statistic Maintenance dialog box. To accept the updates, click OK and all updates are written to the session database.

See 8.4.1 for an example of a derived statistics definition file.

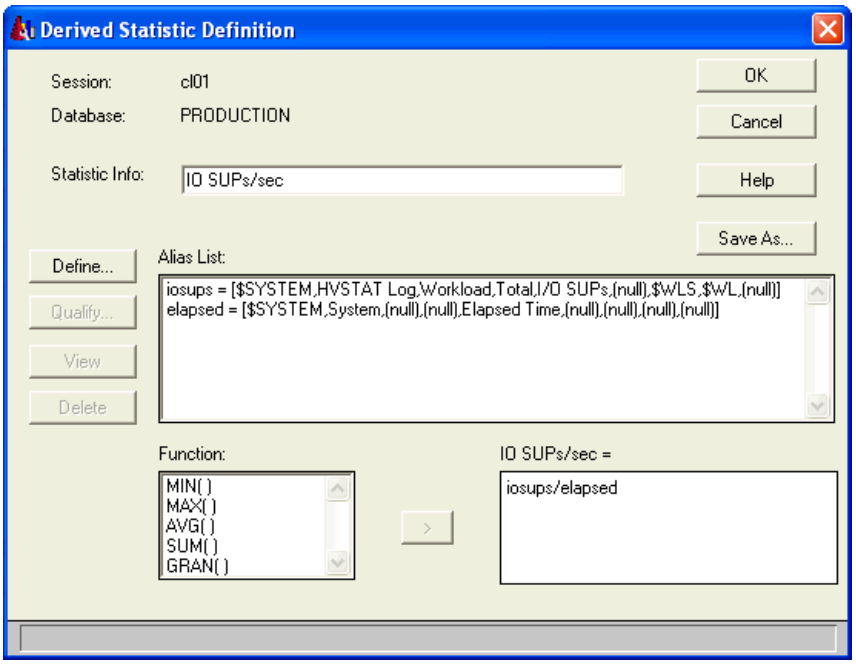
8.3.7. Exporting Derived Statistic Definitions

To export all derived statistic definitions, click the Export button on the Derived Statistic Maintenance dialog box (see 8.3.1). When the Save File dialog box is displayed, select or enter a file that will contain a derived statistic definition for all derived statistics listed in the Derived Statistic Maintenance dialog box.

See 8.4.1 for an example of a derived statistics definition file.

8.3.8. Displaying or Modifying a Derived Statistic Definition

To display or modify a derived statistic definition, select the statistic from the Derived Statistics list box in the Derived Statistic Maintenance dialog box (see 8.3.1) and click Open. You can also double-click the statistic in the Derived Statistics list box. The Derived Statistic Definition dialog box is displayed. The definition information for the selected derived statistic is displayed in the dialog box.



Using this dialog box, you can modify the Statistic Info, the Alias List, and the Formula edit box. For a description of these fields, see 8.3.3.

When you click OK, your changes are accepted and you return to the Derived Statistics Maintenance dialog box (see 8.3.1). Your changes are not actually made in the database until you click OK in the Derived Statistics Maintenance dialog box.

8.3.9. Deleting a Derived Statistic Definition

To delete a derived statistic definition from a database, select a derived statistic from the Derived Statistics list box in the Derived Statistics Maintenance dialog box (see 8.3.1).

When you click Delete, all of the derived statistic definitions are searched for references to the derived statistic you want to delete. If a reference is found, a warning dialog box informs you that you are deleting a derived statistic that is referenced in another derived statistic. You are asked if you still want to delete the statistic.

When the derived statistic definitions are searched and no references are found to the statistic you want to delete, an alert box appears to verify that you have not inadvertently pressed the Delete button.

When you click Yes in this alert box, the statistic is removed from the Derived Statistics list box in the Derived Statistics Maintenance dialog box. To delete the derived statistic from the database, you must click OK.

Note: *If you are deleting a derived statistic that also has a threshold definition, the threshold is also deleted.*

You can cancel the deletion of a derived statistic either by clicking the No button in the alert box or by clicking Cancel in the Derived Statistics Maintenance dialog box (see 8.3.1).

8.4. Maintaining Derived Statistics with tqzip

The **tqzip** program allows you to perform additional tasks associated with maintaining the derived statistics. Using **tqzip** command line options, you can perform the following tasks:

- Unload derived statistic definitions contained in the database to a file (see 8.4.1). You can use this file to view the definitions and update them by using a text editor. You can then reload the definitions.
- Load derived statistic definitions from a specified file into the database (see 8.4.2).

8.4.1. Unloading Derived Statistic Definitions

The **tqzip** program allows you to export all the derived statistic definitions currently contained in the performance database into a file. You can use this file to view the derived statistic definitions and update them by using a text editor. You can then reload the derived statistic definitions by using the -l and -s options on the **tqzip** processor call (see 8.4.2).

Note: *It is recommended that you use the Administration utility to create and update your derived statistic definitions. Using the* ***tqzip*** *processor call options to unload and load a derived statistics file provides an alternative way to update definitions.*

To unload derived statistic definitions with **tqzip**, use the following processor call format:

```
@tqzip [-d,databasename,] -u,-s,derived-file
```

where:

-d specifies the database that **tqzip** uses. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-u specifies that an unload operation is to be performed by **tqzip**. The type of unload operation is “derived statistics” as indicated by the -s option.

-s unloads the derived statistic definitions from the database to the file specified by the *derived-file* parameter.

Derived Statistic Definition File Format

The derived statistic definitions appear in the following format in the output file created by **tqzip**. If you update the derived statistic definition file, you should ensure that the format of the file remains consistent with what was output by the -u and -s options. The following example shows how the definitions appear in the file.

```
BEGIN DERIVED STATISTICS unloaded on Fri Jan 13 17:46:30 2012          A
#                                                                      B
[Statistic Identifier]                                                 C
  category group : "System Log"
  category       : "Workload"
  subcategory    : "Demand"
  statistic      : "Population (open runs)"
[Informational Text]                                                   D
  "Population (open runs)"
[Formula]   ('@' is used as the newline character)                     E
  "active/elapsed"
[Aliases]                                                              F
  alias name       : "active"                                          G
  used in formula  : "Y"                                               H
    system         : "$SYSTEM"                                         I
    category group : "System Log"
    category       : "Workload"
    subcategory    : "Demand"
    statistic      : "Active Time"
    appl group     : ""
    workload set   : "$WLS"
    workload       : "$WL"
    resource       : ""
  alias name       : "elapsed"                                         J
  used in formula  : "Y"
    system         : "$SYSTEM"
    category group : "System"
    category       : ""
    subcategory    : ""
    statistic      : "Elapsed Time"
    appl group     : ""
    workload set   : ""
    workload       : ""
    resource       : ""
#                                                                      K
END DERIVED STATISTICS                                                 L
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A The derived statistics output file starts with a BEGIN DERIVED STATISTICS header line. This line must not be changed or deleted.

B A blank line preceded by a pound sign (#) must follow the header line.

C This is the first line of a derived statistic definition. Each derived statistic definition consists of four sections: [Statistic Identifier], [Informational Text], [Formula], and [Aliases].

This line begins the statistic identifier section. The statistic identifier specifies the name of the derived statistic and shows where it is stored in the database. On subsequent lines following the section title are the key names for the derived statistic. Each of the category group, category, subcategory, and statistic key names must appear on a line by themselves and the actual values must be enclosed in double quotation marks (" ").

D This line begins the informational text section. Descriptive text about the derived statistic follows on subsequent lines. The text must be enclosed in double quotation marks (" ").

E This line begins the formula section. The formula for the derived statistic follows on subsequent lines. It must be enclosed in double quotation marks (" ").

F This line begins the aliases section that defines the aliases associated with the derived statistic. Each part of the alias definition must appear on a line by itself. The actual values must be enclosed in double quotation marks (" ").

G This line specifies the alias name.

H The **Y** indicates that the alias is used in the formula for the derived statistic.

If the alias is not used in the formula, an **N** appears.

I This line begins the parameter hierarchy used to define the alias. Each key name in the hierarchy must be on a line by itself. See 8.2.2 for a description of the $SYSTEM, $WLS, and $WL references in the parameter hierarchy.

J This line begins the second alias definition associated with this derived statistic.

If additional aliases exist for a derived statistic, they would be listed in this format.

K A blank line preceded by a pound sign (#) ends the derived statistic definition.

Additional derived statistic definitions can follow. You can add additional derived statistic definitions by duplicating the preceding lines and modifying the fields within the double quotation marks (" ").

L The last line of the thresholds file must be the END DERIVED STATISTICS line.

8.4.2. Loading Derived Statistic Definitions

You can reload the performance database with a new or updated set of derived statistic definitions by using the -l and -s options on the **tqzip** processor call. If you update the derived statistic definition file, you should ensure that the format of the file remains consistent with the derived statistic definition file created by using the -u and -s options (see 8.4.1).

Notes:

- *If you are using derived statistics for the workload performance reports, you will need to load the derived statistics before data values are available. See A.8 for information on workload performance derived statistics.*
- *When loading derived statistic definitions, the **tqzip** program deletes any existing derived statistic definitions that are not contained in the file being loaded. Use the -e option on **tqzip** if you do not want existing definitions deleted.*

To load derived statistic definitions with **tqzip,** use the following processor call format:

```
@tqzip [-d,databasename,] -l,-s,derived-file [-e]
```

where:

-d specifies the database that **tqzip** uses. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-l specifies that a load operation is to be performed by **tqzip**. The type of load operation is “derived statistics” as indicated by the -s option.

-s loads the derived statistics definitions into the database from the file specified by the *derived-file* parameter. The *derived-file* is the name of a derived statistics definition file and must be in the same format as the file produced by the -u and -s options of **tqzip** (see 8.4.1).

-e loads the new derived statistic definitions from the derived statistic definition file without first deleting the existing definitions in the database. If you do not specify the -e option, all existing definitions are deleted from your database before the new ones are loaded.

Example

The following shows an example of loading derived statistic definitions from the file dfile.

```
>@tqzip -l,-s,dfile
>TQZIP 7R4G (011312 11:04:45) Fri Jan 13 14:26:04 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>
>Using database TQDATA:PRODUCTION
>
>... loading derived statistics from <dfile>
>
>5 derived statistics were loaded.
>
>End TQZIP
```

Section 9
Customizing Data Collection Probes

9.1. Introduction

TeamQuest Baseline and TeamQuest Online collect data from various sources on your OS 2200 system. TeamQuest Online is released with a single probe for SIP data collection that executes as a separate batch run on your system. TeamQuest Baseline is released with probes for SIP, system activity monitor, MIPS Metering, ODTP, TPM log, IOTRACE file, MAPPER log, and system log data. Each of these probes executes as a separate batch run on your system.

The TeamQuest Baseline and TeamQuest Online probes store data into the performance database. The default database is found on the Database setting in the tq-ini element of the product installation file (usually SYSLIB*TQ-BASELINE or SYSLIB*TQ-ONLINE). The default setting is PRODUCTION. (See 10.4 for information on creating additional databases.) You must activate data collection before you can use the TeamQuest View software to monitor or analyze your system activity.

9.2. tqctl Runstream

A control runstream is provided in SYSLIB*RUN$.TQCTL. You can start each probe separately or you can use this runstream to start the data collection probes. For TeamQuest Baseline, the **tqctl** runstream also starts the **tqalm** alarm handler program and the **tqagg** data aggregation utility. Before starting **tqctl**, be sure to terminate all of the runs that it will start (that is, **tqalm**, **tqagg**, and the data collection runs).

The **tqctl** runstream should be automatically started or scheduled after each system reboot. Data will not be collected if the probes are not allowed to run as required on your system.

9.3. SIP Data Collection

TeamQuest Baseline and TeamQuest Online contain a SIP data collection runstream called **tqbsp**. The **tqbsp** runstream uses a run identifier of **tqbsp** and executes the **tqbsp** program. The **tqbsp** program collects SIP data and stores it in the performance database. The data collected by **tqbsp** is also available for real-time system monitoring.

The **tqbsp** program uses the ER SYSBAL$ interface for collecting SIP data from the OS 2200 system. You should allow the **tqbsp** runstream to execute continuously as a background run on your system. This ensures that your performance database contains a complete set of data for each day and that the real-time monitor can access the most current data.

9.3.1. tqbsp Processor Call Statement

The **tqbsp** runstream contains the **tqbsp** processor call statement, which controls SIP data collection.

The **tqbsp** processor call statement specifies data collection parameters including input parameter table name, database name, and aggregation set names. Normally, you use the default values for these parameters as provided in the **tqbsp** runstream. To modify any of the defaults, you need to update the **tqbsp** runstream in SYSLIB*RUN$.

The **tqbsp** processor call has the following format:

```
@tqbsp [-r,] -i,parameter-table,
       [-d,database-name,]
       [-a,aggset1,[-a,aggset2,]]
       END
```

where:

-r executes **tqbsp** in real-time mode. The account for the **tqbsp** run must have a maximum real-time level greater than zero.

-i specifies the location of the input parameter table name to be used by the **tqbsp** probe. For the **tqbsp** probe, the *parameter-table* is the element SYSLIB*TQ-BASELINE.SIPINPUTTBL. This is a required parameter on the **tqbsp** call line.

-d specifies the name of the database to be used for data collection. The *database-name* parameter is the name of an existing database on your system. If this parameter is not specified, the Database setting in the tq-ini element is used. Make sure that all programs called in a runstream use the same database name.

-a specifies the aggregation set to be used for data collection. The *aggset1* and *aggset2* parameters are the names of aggregation sets defined in the *database-name*. The *aggset1* and *aggset2* parameters are case-sensitive.

All aggregation sets are optional on the call line. If no aggregation sets are specified, the Aggset1 and Aggset2 settings of the Aggregation Sets section in the tq-ini element are used. See 2.2.1 for a description of aggregation sets.

Note: *Parameters passed to a program from the Exec call line are in lowercase. Some parameters on the program call statement are case-sensitive. These parameters should appear on the second and subsequent lines of the processor call.*

The parameters specified on the processor call statement must be separated by commas. Do not split a single parameter across two lines. No continuation character is allowed when using multiple lines. The processor call statement must be terminated with the END keyword.

9.3.2. Terminating Data Collection

To terminate the **tqbsp** run from a demand run that has full console mode capabilities or greater, enter

```
@@CONS TQBSP END
```

To terminate the **tqbsp** run from the system console, enter

```
TQBSP END
```

9.3.3. Abnormal Termination of Data Collection

If **tqbsp** terminates in error due to a maximum time contingency, it restarts automatically. If SIP is turned off at the console, **tqbsp** turns SIP back on automatically and uses the ER SYSBAL$ interface upon restarting to continue data collection.

If **tqbsp** terminates and does not restart, try to resolve the problem as soon as possible and restart **tqbsp**. Data is collected only when **tqbsp** is running.

9.3.4. SIP File Input

With the **tqbsp** program, you can load SIP data directly from a SIP file into a performance database. This method enables you to load SIP data from another system or to update the performance database with SIP data that was generated at a time when **tqbsp** was not running on your system.

To load SIP data directly from SIP files, you must specify three additional parameters on the **tqbsp** processor call statement. (See 9.3.1 for the format of the processor call statement.) The format for these parameters is as follows:

```
-a,aggset1,
-f,sipfile(n).,[-f,sipfile(n+1)., ...,]
-s,system-name,
```

where:

-a specifies the aggregation set to be used for data collection. The *aggset1* parameter is the name of the aggregation set defined in the *database-name*. The *aggset1* parameter is case-sensitive. Only one aggregation set is allowed.

-f specifies a SIP file to be processed. *sipfile* is the name of the SIP file. *n* is the file cycle number. When processing multiple system log files, multiple -f options are needed, one for each cycle to be processed. All cycles must be in sequential order and from the same SIP session. You can process SIP tape files that have been previously assigned to your run by specifying the file name used to assign the tape.

-s specifies the *system-name* that identifies the system source of the SIP data. It is used to determine where to store the data in the performance database. The *system-name* parameter is case-sensitive.

The **tqbsp** -d and -a parameters (see 9.3.1) are also used for SIP file input. Only one aggregation set (-a) can be specified when using SIP file input and the -a parameter is required on the **tqbsp** processor call. If you do not want **tqbsp** to load the SIP data into the version of the performance database specified on the Database setting in the tq-ini file, use the -d parameter to specify an alternate database version.

Note: *Before loading data into an alternate version of the performance database, you must use the* ***tqdbu*** *utility to create this alternate version (see 10.4).*

The **tqbsp** program processes each SIP file, storing data points in the performance database as specified by the aggregation set parameter (-a). If the SIP file terminates prematurely (that is, if the last block is not a SIP summary block), the last data point is not written to the database. If the last SIP block of a file is a SIP EOF and another SIP file is specified on the **tqbsp** processor call statement, the next SIP file is processed.

The granularity of the input SIP file must not be greater than the granularity of the aggregation set specified as the -a parameter on the **tqbsp** processor call statement (see 9.3.1).

9.4. System Activity Monitor Collection

Note: *The System Activity Monitor (**tqsam**) program is released with TeamQuest Baseline only. It is not available with TeamQuest Online.*

TeamQuest Baseline contains a system activity monitor runstream called **tqsam**. The **tqsam** runstream uses a run identifier of **tqsam** and executes the **tqsam** program. The **tqsam** program collects the most current information available on system run and transaction activity on your OS 2200 system.

The **tqsam** program makes use of a set of Exec callable interface routines that return the most up-to-date information on all active runs and transactions on your OS 2200 system. System activity is monitored by the **tqsam** program at every sample interval and data is stored in several tables. Information on all active runs is stored in the Runs table and transaction information is optionally collected and stored in the Transactions table. See Sections 2 and 11 for more information on the collection of table data.

9.4.1. tqsam Processor Call Statement

The **tqsam** runstream contains the **tqsam** processor call statement that controls the system activity monitor routines to collect information on all active runs and the longest running transactions on your system.

The **tqsam** processor call statement includes options to collect statistics on transactions, to run the probe in real time, to specify the database name, and to specify the maximum number of longest running transactions to monitor. The **tqsam** runstream can be edited and tailored to your site-specific needs.

The **tqsam** processor call has the following format:

```
@tqsam [-r,] [-t,]
       [-d,database-name,]
       [-n,#transactions,]
       END
```

where:

-r executes **tqsam** in real-time mode. The account for the **tqsam** run must have a maximum real-time level greater than zero. By running in real-time mode, you can prevent the probe from missing data as a result of being swapped out during scheduled sample periods. The default value is to not run in real-time mode.

-t specifies that statistics on transactions should be collected.

If you are not running TIP or do not want TIP statistics collected, do not use the -t option. This will reduce processing time and database storage overhead. The default value is not to collect TIP statistics.

-d specifies the name of the database to be used for data collection. The *database-name* parameter is the name of an existing database on your system. If this parameter is not specified, the Database setting in the tq-ini file is used. Make sure that all programs called in a runstream use the same database name.

-n specifies the number of active, longest-running transactions that should be monitored, stored, and reported in the table report. This option can be used with the -t option. The #*transactions* parameter can be any number from 1 to 20. The default value is to collect information on the 20 longest running, active transactions.

Note: *Parameters passed to a program from the Exec call line are in lowercase. Some parameters on the program call statement are case-sensitive. These parameters should appear on the second and subsequent lines of the processor call.*

The parameters specified on the processor call statement must be separated by commas. Do not split a single parameter across two lines. No continuation character is allowed when using multiple lines. The processor call statement must be terminated with the END keyword.

The **tqsam** probe has a default collection period, retention period, and sample interval. See Section 11 for more information on maintaining table data. The defaults are as follows:

collection period
: The collection period indicates a time range that the **tqsam** probe, when running, is to sample system activity. By default, **tqsam** will continuously collect data.

sample interval
: The sample interval (in seconds) indicates how often the **tqsam** probe is to sample system activity within a collection period. The default sample interval is 30 seconds.

retention period
: A retention period represents how long the data is maintained in the database. A retention period called Primary exists for each table and has a default retention period of 24 hours.

9.4.2. Terminating Data Collection

To terminate the **tqsam** run from a demand run that has full console mode capabilities or greater, enter

```
@@CONS TQSAM END
```

To terminate the **tqsam** run from the system console, enter

```
TQSAM END
```

9.4.3. Abnormal Termination of Data Collection

If **tqsam** terminates in error due to a maximum time contingency, it restarts automatically. If **tqsam** terminates and does not restart, try to resolve the problem as soon as possible and restart **tqsam**. Data is collected only when **tqsam** is running.

9.4.4. Workload Characteristics for the System Activity Monitor Probe

Note: *Workload evaluation is performed each time the data is retrieved for reporting by TeamQuest Performance Software Administration. This provides for dynamic workload classification as the workload definition can be changed using TeamQuest Performance Software Administration and reflected in the **tqsam** reports. This allows you to dynamically fine-tune a workload definition that might also be used in the **tqsys** probe.*

The **tqsam** probe has a unique set of characteristics that are used for its workload definitions. You use these characteristics when you create the workload definitions using TeamQuest Performance Software Administration (see 5.3). The following characteristics define the **tqsam** workloads:

Name	Characters	Type/Length	Operations
Run_ID	A–Z, 0–9	string,6	=,!=
Program	A–Z, 0–9,-,$	string,12	=,!=
Transaction	A–Z, 0–9	string,12	=,!=
Account	A–Z, 0–9,-, period(.)	string,12	=,!=
Type	a–z, -	string,10	=,!=

The allowable values for Type in the Runs table are

demand	(demand)
demand-rt	(demand real-time)
batch	(batch)
batch-rt	(batch real-time)
tip	(TIP)

The allowable values for Type in the Transactions table are

self-d	(self-destructive)
self-i	(self-initializing)
reent	(reentrant)
onl-bat	(online-batch)
hv-tip	(high-volume TIP)

9.4.5. tqsam Table Data

The **tqsam** probe collects and stores data in two tables: the Runs table and, optionally, the Transactions table. The Runs table contains data collected on behalf of demand and batch run activity. When the -t option is specified on the **tqsam** processor call line, the Transactions table contains data collected on the longest running transactions that are active. Information is updated in these tables at the interval rate specified for each table.

9.4.5.1. Runs Table Information

The Runs table contains information on currently active system demand and batch run activity. This information is reported in a tabular column format and is displayed by TeamQuest View. See A.9.1 for a list of the information stored in the Runs table by **tqsam**.

9.4.5.2. Transactions Table Information

The Transactions table contains information on the top 20 longest running and active transactions. By viewing the longest running transactions, you can identify those transactions that may be held up for some particular reason. This information is reported in a tabular column format and is displayed by TeamQuest View. See A.9.2 for a list of the information stored in the Transactions table by **tqsam**.

9.5. MIPS Metering Data Collection

TeamQuest Baseline contains a MIPS Metering data collection runstream called **tqmips**. The **tqmips** runstream uses a run identifier of **tqmips** and executes the **tqmips** program. The **tqmips** program reads from the dedicated Capacity on Demand (COD) audit trail or the system log file and collects MIPS Metering data for a COD partition and stores it in the performance database. The data collected by **tqmips** is also available for real-time system monitoring.

You should allow the **tqmips** runstream to execute continuously as a background run on your system. This ensures that your performance databases contain a complete set of data for each day and that the real-time monitor can access the most current data.

9.5.1. tqmips Processor Call Statement

The **tqmips** runstream contains the **tqmips** processor call statement, which controls the collection of MIPS metering data collection.

There are two trails that contain the same MIPS information. By default, the **tqmips** probe reads and processes from the dedicated COD audit trail. The COD audit trail contains only COD MIPS related data. MIPS data is also contained in the system log, but it is much less efficient to process since it contains data other than COD MIPS data. For selection of the audit trail number, see the -n option in the format description.

The **tqmips** processor call statement specifies data collection parameters including input parameter table name, database name, and aggregation set names. Normally, you use the default values for these parameters as provided in the **tqmips** runstream. To modify any of the defaults, you need to update the **tqmips** runstream in SYSLIB*RUN$.

The **tqmips** processor call has the following format:

```
@tqmips [-r,] -i,parameter-table,
        [-d,database-name,]
        [-a,aggset1,[-a,aggset2,]]
        [-n,audit-trail-number]
        END
```

where:

-r executes **tqmips** in real-time mode. The account for the **tqmips** run must have a maximum real-time level greater than zero. The -r option is recommended so that the **tqmips** probe has the best opportunity to report data without missing a sample as the data is being written to the COD audit trail and system log. The default is to not run in real-time.

-i specifies the location of the input parameter table name to be used by the **tqmips** probe. For the **tqmips** probe, the *parameter-table* is the element SYSLIB*TQ-BASELINE.MIPSINPUTTBL. This is a required parameter on the **tqmips** call line.

-d specifies the name of the database to be used for data collection. The *database-name* parameter is the name of an existing database on your system. If this parameter is not specified, the Database setting in the tq-ini element is used. Make sure that all programs called in a runstream use the same database name.

-a specifies the aggregation set to be used for data collection. The *aggset1* and *aggset2* parameters are the names of aggregation sets defined in the *database-name*. The *aggset1* and *aggset2* parameters are case-sensitive. All aggregation sets are optional on the call line. If aggregation sets are not specified, the Aggset1 and Aggset2 settings of the Aggregation Sets section in the tq-ini element are used. See 2.2.1 for a description of aggregation sets.

-n specifies which audit trail number to use. This option should only be used to read MIPS data from a trail other than the dedicated COD audit trail. To read from the system log file, use -n,*audit-trail-number* specification where *audit-trail-number* is the system log trail number. Processing from the COD audit trail is much more efficient than using the -n option.

Note: *Parameters passed to a program from the Exec call line are in lowercase. Some parameters on the program call statement are case-sensitive. These parameters should appear on the second and subsequent lines of the processor call.*

The parameters specified on the processor call statement must be separated by commas. Do not split a single parameter across two lines. No continuation character is allowed when using multiple lines. The processor call statement must be terminated with the END keyword

9.5.2. Terminating Data Collection

To terminate the **tqmips** run from a demand run that has full console mode capabilities or greater, enter

```
@@CONS TQMIPS END
```

To terminate the **tqmips** run from the system console, enter

```
TQMIPS END
```

9.5.3. Abnormal Termination of Data Collection

If **tqmips** terminates in error due to a maximum time contingency, it restarts automatically.

If **tqmips** terminates and does not restart, try to resolve the problem as soon as possible and restart **tqmips**. Data is collected only when **tqmips** is running.

9.5.4. Trail Selection By Date Range or Relative Time

With the **tqmips** program, you can load data from COD audit trails or from system log files into a performance database by specifying a date or relative time range. This enables you to fill in gaps in the performance database with MIPS data that was collected at a time when **tqmips** was not running on your system.

When loading data using a date range, **tqmips** searches the system and processes all trails that contain information in the date range. You must specify up to 3 additional parameters on the **tqmips** processor call statement. See 9.5.1 for the format of the processor call statement. The format for the date range parameters is

```
-b,mmddyy[yy],
-e,mmddyy[yy],
[-n,audit-trail-number].
```

where:

-b specifies the month, day, and year of the beginning date used in the log file selection.

-e specifies the month, day, and year of the ending date used in the log file selection.

-n specifies which audit trail number to use. This option should only be used to read MIPS data from a trail other than the dedicated COD audit trail. To read from the system log file, use -n,*audit-trail-number* specification where *audit-trail-number* is the system log trail number. Processing from the COD audit trail is much more efficient than using the -n option.

If a 2-digit year is specified, the digits 00 through 27 are interpreted as the years 2000 through 2027 and the digits 28 through 99 are interpreted as the years 1928 through 1999.

You can also set a relative time option when selecting the log entries to process. The format of the relative time option is

```
-t,relative-hour
[-n,audit-trail-number].
```

where:

-t specifies a relative hour when selecting log entries to process. The valid range is 1 to 24. For example, -t,8 selects and processes the last 8 hours of log data relative to when the **tqtmips** program was started.

Note: *A value greater than 24 selects and processes all of the previous days' data.*

-n specifies which audit trail number to use. This option should only be used to read MIPS data from a trail other than the dedicated COD audit trail. To read from the system log file, use -n,*audit-trail-number* specification where *audit-trail-number* is the system log trail number. Processing from the COD audit trail is much more efficient than using the -n option.

9.5.5. System Log File Input

Note: *Specific COD audit trails cannot be listed for processing since the COD entries within the COD audit trails do not contain local timestamps necessary to store into the TeamQuest performance database. As a result, only system log files can be input for processing as described here.*

To load data directly from specific log files, you must specify two additional parameters on the **tqmips** processor call statement. See 9.5.1 for the format of the processor call statement. The format for these parameters is

```
-f,logfile(n).,[-f,logfile(n+1)., ...,]
-s,system-name,
-n,audit-trail-number.
```

where:

-f specifies the names of the system log files to be processed. *logfile* is the name of the system log file. *n* is the file cycle number. When processing multiple log files, multiple -f options are needed, one for each cycle to be processed. All files must be in sequential order.

-s specifies a system name. The *system-name* parameter specifies the name of the system source of the system log files. It determines where to store the data in the performance database. The *system-name* parameter is case-sensitive.

-n specifies the audit trail number of the system log (-n,10 unless the system log is locally configured to a different *audit-trail-number*).

The **tqmips** -d and -a parameters are also required for log file input. Only one -a aggregation set parameter can be specified when using the -f parameter. If you do not want **tqmips** to load the log data into the version of the performance database specified on the Database setting in the tq-ini file, use the -d parameter to specify an alternate database version.

Note: *Before loading data into an alternate version of the performance database, you must use the* ***tqdbu*** *utility to create this alternate version (see 10.4).*

If the -b and -e parameters are used with the -f parameter, **tqmips** processes the information in the specified log files for the date range specified on the -b and -e parameters.

The **tqmips** program processes each log file, storing data points in the performance database as specified by the -a aggregation set parameter.

9.6. Open Distributed Transaction Processing Collection

Note: *The Open Distributed Transaction Processing (**tqodtp**) probe is released with TeamQuest Baseline only. It is not available with TeamQuest Online.*

TeamQuest Baseline contains an Open Distributed Transaction Processing (ODTP) monitor runstream called **tqodtp**. The **tqodtp** runstream uses a run identifier of **tqodtp** and executes the **tqodtp** program.

The **tqodtp** program collects the most up-to-date information available on an OS 2200 open, standard-compliant system environment. Statistics on overall system, system clients, system server programs, and service routines are collected. The **tqodtp** program uses a monitoring interface supplied with the Unisys OLTP-TM2200 product. ODTP data is monitored by the **tqodtp** program at every sample interval and data is stored in several tables. See Sections 2 and 11 for more information on the collection of table data.

9.6.1. tqodtp Processor Call Statement

The **tqodtp** runstream contains the **tqodtp** processor call statement that controls the collection of information on active client programs, server programs, and service routines in an ODTP environment.

The **tqodtp** processor call has the following format:

```
@tqodtp [-r,]
        [-d,database-name,]
        END
```

where:

-r executes **tqodtp** in real-time mode. The account for the **tqodtp** run must have a maximum real-time level greater than zero. By running in real-time mode, you can prevent the probe from missing data as a result of being swapped out during scheduled sample periods. The default value is to not run in real-time mode.

-d specifies the name of the database to be used for data collection. The *database-name* parameter is the name of an existing database on your system. If this parameter is not specified, the Database setting in the tq-ini file is used. Be sure that all programs called in a runstream use the same database name.

Note: *Parameters passed to a program from the Exec call line are in lowercase. Some parameters on the program call statement are case-sensitive. These parameters should appear on the second and subsequent lines of the processor call.*

The parameters specified on the processor call statement must be separated by commas. Do not split single parameters across two lines. No continuation character is allowed when using multiple lines. The processor call statement must be terminated with the END keyword.

The **tqodtp** probe has a default collection period, retention period, and sample interval. See Section 11 for more information on maintaining table data. The defaults are

Collection period
: The collection period indicates a time range that the **tqodtp** probe, when running, is to sample system activity. By default, **tqodtp** will continuously collect data.

Sample interval
: The sample interval (in seconds) indicates how often the **tqodtp** probe is to sample system activity within a collection period. The default sample interval is 1 minute.

Retention period
: A retention period represents how long the data is maintained in the database. A retention period called Primary exists for each table and has a default retention period of 24 hours.

9.6.2. Terminating Data Collection

To terminate the **tqodtp** run from a demand run that has full console mode capabilities or greater, enter

```
@@CONS TQODTP END
```

To terminate the **tqodtp** run from the system console, enter

```
TQODTP END
```

9.6.3. Abnormal Termination of Data Collection

If **tqodtp** terminates in error due to a maximum time contingency, it restarts automatically. If **tqodtp** terminates and does not restart, try to resolve the problem as soon as possible and restart **tqodtp**. Data is collected only when **tqodtp** is running.

9.6.4. tqodtp Table Data

The **tqodtp** probe collects and stores data into several tables. The **tqodtp** probe monitors ODTP information using an interface supplied with the OLTP-TM2200 product. The **tqodtp** probe monitors the information at each sample interval. The ODTP configuration parameter, STATS_LEVEL, must be set to one or greater for the data to be available for collection by the **tqodtp** probe. For additional information on configuring STATS_LEVEL, see the *Distributed Processing Middleware Open Distributed Transaction Processing Administration Guide Volume 1: Installation, Configuration, and Ongoing Administration*.

9.6.4.1. System Table Information

The System table contains information specific to the ODTP environment. This information is reported in a tabular column format and is displayed by TeamQuest View. See A.10.1 for a list of the information stored in the System table by **tqodtp**.

9.6.4.2. Server Table Information

The Server table contains information specific to the ODTP environment. This information is reported in a tabular column format and is displayed by TeamQuest View. See A.10.2 for a list of the information stored in the Server table by **tqodtp**.

9.6.4.3. Client Table Information

The Client table contains information specific to the ODTP environment. This information is reported in a tabular column format and is displayed by TeamQuest View. See A.10.3 for a list of the information stored in the Client table by **tqodtp**.

9.6.4.4. Service Table Information

The Service table contains information specific to the ODTP environment. This information is reported in a tabular column format and is displayed by TeamQuest View. See A.10.4 for a list of the information stored in the Service table by **tqodtp**.

9.7. Real-Time TPM and TPM Log Data Collection

Note: *The real-time TPM (RTPM) and TPM log probes are released with TeamQuest Baseline only. These probes are not available for TeamQuest Online.*

The **tqrtpm** and **tqtpm** log probes report the TIP/HVTIP statistics from the TIP Performance Monitor (TPM) audit trail. The **tqrtpm** probe runs continuously to report the data in a near real-time mode. The **tqtpm** probe reports data from the previous day, while the **tqrtpm** probe reads the TPM audit trail as new data is written.

Both probes can be used on your system. The **tqrtpm** probe reports data into one aggregation set to be viewed continuously and provides the ability to view the TIP/HVTIP transactions. Normally, the **tqrtpm** probe reports 1-minute data into the 1-minute aggregation set which is not aggregated by **tqagg**. The **tqtpm** probe can be configured to start after midnight, process the previous day's data, and populate the 10-minute aggregation set that is aggregated when **tqagg** runs.

Both probes process TPM workload data based on the same workload sets. For the **tqrtpm** probe, the workload sets can optionally be specified on the call line to limit the amount of workload processing required and reduce processing overhead. Depending on the volume of transactions that occur, the **tqrtpm** probe may fall behind while reporting data in near real-time. For this reason, it is recommended that the **tqrtpm** probe be run with the -r option to enable it to run at a real-time priority.

Both probes require that your TPM audit trail be configured to mass storage or moved to mass storage prior to starting. These probes report on information from the TPM TIP termination records type 13, subtype 3, generated when TIP is configured in the Exec. By default, audit trail #17 is referenced for TPM log file selection. For example, the ACI SYS$*ACI$17 and SYS$*F017L1 file cycles. The audit trail number can be specified on the **tqrtpm** or **tqtpm** processor call statement.

By default, the **tqtpm** program processes all cycles of the TPM log files from the previous day. During the tqinit process at installation time, you are prompted for the time that the **tqtpm** runstream starts each day. The **tqtpm** probe should run each day to ensure that your performance database is kept up-to-date.

Both probes also analyze a common format of the user area common to some sites using the Logic and Information Network Complier (LINC 1100). The LINC user area format is the same as that described in 9.7.6.

9.7.1. tqrtpm Processor Call Statement

The **tqrtpm** runstream contains the **tqrtpm** processor call statement, which is used to start the **tqrtpm** probe data collection.

The **tqrtpm** processor call statement specifies data collection parameters including input parameter table and optionally the database name, aggregation set name, workload set name, and audit trail number. To modify any of these defaults, you need to update the **tqrtpm** runstream in SYSLIB*RUN$.

It is recommended that the **tqrtpm** probe be run at real-time by specifying the -r option, otherwise the probe may fall behind while reporting data.

The **tqrtpm** probe can interpret the user area of the TPM termination records in the TPM audit trail. See 9.7.6 for the LINC 1100 user formatted area that the probe can process when the -u option has been specified. The default is to not process the user area.

The **tqrtpm** processor call has the following format:

```
@tqrtpm [-r] [-u] [-z,] -i,parameter-table,
        [-d,database-name,]
        [-a,aggset1]
        [-w,workload-set-name,] [-w,workload-set-name,]
        [-n,audit-trail-number,]
        END
```

where:

-r executes **tqrtpm** in real-time mode. The account for the **tqrtpm** run must have a maximum real-time level greater than zero. The -r option is recommended so the **tqrtpm** probe run has the best opportunity to report data as it is being written to the TPM audit trail. The default is to not run in real-time.

-u processes the user area of TPM termination records using the LINC format described in 9.7.6.

-z is used to register the **tqrtpm** probe in the database.

-i specifies the location of the input parameter table name to be used by the **tqrtpm** probe. For the **tqrtpm** probe, the parameter-table is the element SYSLIB*TQ-BASELINE.TPMINPUTTBL. This is a required parameter on the **tqrtpm** call line.

-d specifies the name of the database to be used for data collection. The *database-name* parameter is the name of an existing database on your system. If this parameter is not specified, the Database setting in the tq-ini element is used. Make sure that all programs called in a runstream use the same database name.

-a specifies the aggregation set to be used for data collection. The *aggset1* parameter is the name of an aggregation set defined in the *database-name*. The *aggset1* parameter is case-sensitive and only one aggregation set is allowed.

The aggregation set is optional on the call line. If an aggregation set is not specified, the Aggset1 setting of the Aggregation Sets section in the tq-ini element is used. (See 2.2.1 for a description of aggregation sets.)

-w specifies the workload set names to be used in data collection. If a workload set name is not supplied, all active workload sets for the source data type "hvslog" are used.

-n specifies the TPM audit trail number. By default, this value is 17 corresponding to SYS$*ACI$17 and the SYS$*F017L1 audit trail file names used in file selection.

Note: *Parameters passed to a program from the Exec call line are in lowercase. Some parameters on the program call statement are case-sensitive. These parameters should appear on the second and subsequent line of the processor call.*

9.7.2. Terminating tqrtpm Data Collection

To terminate the **tqrtpm** run from a demand run that has full console mode capabilities or greater, enter the following:

```
@@CONS TQRTPM END
```

To terminate the **tqrtpm** run from the system console, enter the following:

```
TQRTPM END
```

9.7.3. tqtpm Processor Call Statement

The **tqtpm** runstream contains the **tqtpm** processor call statement, which controls the TPM log data collection.

The **tqtpm** processor call statement specifies data collection parameters including input parameter table, database name (optional), aggregation set name, and audit trail number (optional). Normally, you use the default values for these parameters as provided in the **tqtpm** runstream. To modify any of these defaults, you need to update the **tqtpm** runstream in SYSLIB*RUN$.

The **tqtpm** processor call has the following format:

```
@tqtpm [-u] [-z,] -i,parameter-table,
       [-d,database-name,]
       [-a,aggset1]
       [-n,audit-trail-number]
       END
```

where:

-u processes the user area of TPM termination records using the LINC format described in 9.7.6.

-z is used to register the **tqtpm** probe in the database.

-i specifies the location of the input parameter table name to be used by the **tqtpm** probe. For the **tqtpm** probe, the parameter-table is the element SYSLIB*TQ-BASELINE.TPMINPUTTBL. This is a required parameter on the **tqtpm** call line.

-d specifies the name of the database to be used for data collection. The *database-name* parameter is the name of an existing database on your system. If this parameter is not specified, the Database setting in the tq-ini element is used. Make sure that all programs called in a runstream use the same database name.

-a specifies the aggregation set to be used for data collection. The *aggset1* parameter is the name of an aggregation set defined in the *database-name*. The *aggset1* parameter is case-sensitive.

The aggregation set is optional on the call line. If no aggregation set is specified, the Aggset1 setting of the Aggregation Sets section in the tq-ini element is used. (See 2.2.1 for a description of aggregation sets.)

-n specifies the TPM audit trail number. By default, this value is 17 corresponding to SYS$*ACI$17 and the SYS$*F017L1 audit trail file names used in file selection.

Note: *Parameters passed to a program from the Exec call line are in lowercase. Some parameters on the program call statement are case-sensitive. These parameters should appear on the second and subsequent lines of the processor call.*

9.7.4. TPM Log File Input

With the **tqtpm** program, you can load data from a TPM log file into a performance database by specifying the date range or the specific TPM log file. This enables you to load TPM log data from another system or to update the performance database with TPM log data that was collected at a time when **tqtpm** was not running on your system.

When loading data using a date range, **tqtpm** searches the system and processes all TPM log files that contain information in the date range. You must specify two additional parameters on the **tqtpm** processor call statement. See 9.7.3 for the format of the processor call statement. The format for the date range parameters is

```
-b,mmddyy[yy],
-e,mmddyy[yy],
```

where:

-b specifies the month, day, and year of the beginning date used in the log file selection.

-e specifies the month, day, and year of the ending date used in the log file selection.

If a 2-digit year is specified, the digits 00 through 27 are interpreted as the years 2000 through 2027 and the digits 28 through 99 are interpreted as the years 1928 through 1999.

To load data directly from specific TPM log files, you must specify two additional parameters on the **tqtpm** processor call statement. See 9.7.3 for the format of the processor call statement. The format for these parameters is:

```
-f,tpmlog(n).,[-f,tpmlog(n+1)., ...,]
-s,system-name,
```

where:

-f specifies the names of the TPM log files to be processed. *tpmlog* is the name of the TPM log file. *n* is the file cycle number. When processing multiple TPM log files, multiple -f options are needed, one for each cycle to be processed. All files must be in sequential order.

-s specifies a system name. The *system-name* parameter specifies the name of the system source of the system log files. It determines where to store the data in the performance database. The *system-name* parameter is case-sensitive.

The **tqtpm** -d and -a parameters are also required for TPM log file input (see 9.7.3). Only one aggregation set (-a) can be specified when using the -f parameter. If you do not want **tqtpm** to load the system log data into the version of the performance database specified on the Database setting in the tq-ini file, use the -d parameter to specify an alternate database version.

Note: *Before loading data into an alternate version of the performance database, you must use the* ***tqdbu*** *utility to create this alternate version (see 10.4).*

If the -b and -e parameters are used with the -f parameter, **tqtpm** processes the information in the specified log files for the date range specified on the -b and -e parameters.

The **tqtpm** program processes each system log file, storing data points in the performance database as specified by the aggregation set parameter -a.

You can also set a relative time option when selecting the TPM log entries to process. The format of the relative time option is

```
-t,relative-hour
```

where:

-t specifies a relative hour when selecting TPM log entries to process. The valid range is 1 to 24. For example, -t,8 will select and process the last 8 hours of TPM log data relative to when the **tqtpm** program is started.

9.7.5. Workload Characteristics for the RTPM and TPM Log File Probes

The RTPM and TPM log probes have a unique set of characteristics that are used for its workload definitions. You use these characteristics when you create the workload definitions using TeamQuest Performance Software Administration (see 5.3). The following characteristics define the RTPM and TPM log workloads:

Characteristic Name	Valid Characters	Type and Max Length	Valid Operations
Txn_Code	A–Z, 0–9, ~, ^, %	string, 6	=,!=
PID	0–9	integer, 9	=,<,>,<=,>=,!=
Program_Name	A–Z, 0–9, -, $	string, 12	=, !=
Run_ID	A–Z, 0–9	string, 6	=, !=
Lib_Bank	0–9, -	string, 12	=, !=

If your site is gathering LINC information in the log file user area, you can use the following additional characteristics:

Characteristic Name	Valid Characters	Type and Max Length	Valid Operations
ISPEC_Name	A–Z, 0–9	string, 5	=, !=
Txn_Run_ID	A–Z, 0–9, *	string, 6	=, !=
Appl _Group	0–9	integer, 2	=, <, >, <=, >=, !=

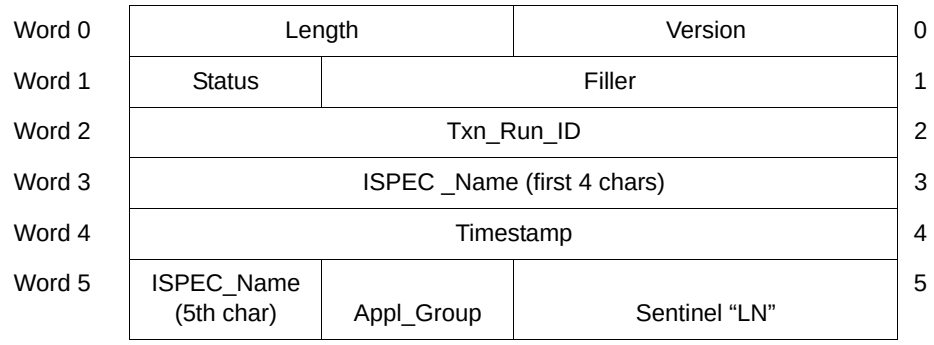
Example

The following example shows a workload definition, using program names to identify system transactions. This workload contains information for system transactions CSF0 through CSF99, XYZ, and SYSERR. By using the Txn_Code=/CSF[0-9] [0-9]?/ construct, data for transactions with identifiers of CSF0 and CSF9 and CSF00 through CSF99 will be included in this workload.

```
Txn_Code={/CSF[0-9][0-9]?/,"XYZ","SYSERR"}
```

9.7.6. Log File User Area

This format of the user area is common to some sites using the Logic and Information Network Compiler (LINC 1100). The following shows the format of the log file user area.



where:

Length	is the length of the user area, currently 6 words (18 bits).
Version	is the version of the user area (18 bits).
Status	is the status of record. A zero value indicates the record is OK (6 bits).
Filler	is not used (30 bits).
Txn_Run_ID	is the unique transaction run-id of this transaction (6 FIELDATA characters).
ISPEC_Name	is the name of the initially called control program. The first four characters are in word 3 with the 5th character in Q1 of word 5. These are ASCII characters.
Timestamp	is the timestamp of when this execution was scheduled.
Appl _Group	is the step control application group number used in this LINC system (9 bits).
Sentinel	indicates the “LN” or LINC format (18 bits).

9.8. IOTRACE Data Collection

Note: *The IOTRACE probe is released with TeamQuest Baseline only. It is not available for TeamQuest Online.*

IOTRACE is an optional feature of SIP that can run independently or with SIP data collection. It gathers detailed information about every I/O operation to tape or mass storage device. This information is written to the IOTRACE data file SYS$*IODATFILE, which is a tape file with standard labels or a mass storage file. Information contained in each trace entry includes the file referenced, device referenced, run-id, program, service times, words transferred, and I/O function.

9.8.1. Data Collection Runstream

TeamQuest Baseline contains a runstream called **tqiot** that processes IOTRACE data collection files. The **tqiot** runstream uses a run identifier of **tqiot** and executes the **tqiot** program. The **tqiot** program retrieves data from the IOTRACE file and stores it in the performance database.

Note: *The **tqiot** runstream is a template runstream that must be updated before it can be used. You can update the template or copy the template to a different file and make updates to the **tqiot** processor call statement as required for your site.*

The **tqiot** program processes the set of IOTRACE files specified on the **tqiot** program call line. The **tqiot** program should run each day that you collect IOTRACE data to ensure that your performance database contains data for each day.

9.8.2. tqiot Processor Call Statement

The **tqiot** runstream contains the **tqiot** processor call statement, which controls the IOTRACE data reduction.

The **tqiot** processor call statement specifies data collection parameters including input parameter table name, database name, aggregation set name, IOTRACE file names, and system name. Normally, you use the default values for these parameters as provided in the **tqiot** runstream. To modify any of the defaults, you need to update the **tqiot** runstream in SYSLIB*TQ-BASELINE.

The **tqiot** processor call has the following format:

```
@tqiot [-z,] -i,parameter-table,
       [-d,database-name,]
       [-a,aggset1,]
        -f,IOTRACE-log(n).,[-f,IOTRACE-log(n+1)., ...,]
        -s,system-name,
        END
```

where:

-z is used to register the **tqiot** probe in the database.

-i specifies the location of the input parameter table name to be used by the **tqiot** probe. For the **tqiot** probe, the *parameter-table* is the element SYSLIB*TQ-BASELINE.IOTINPUTTBL. This is a required parameter on the **tqiot** call line.

-d specifies the name of the database to be used for data collection. The *database-name* parameter is the name of an existing database on your system. If this parameter is not specified, the Database setting in the tq-ini element is used. Make sure that all programs called in a runstream use the same database name.

-a specifies the aggregation set to be used for data collection. The *aggset1* parameter is the name of an aggregation set defined in the *database-name*. The *aggset1* parameter is case-sensitive.

The aggregation set is optional on the processor call. If no aggregation set is specified, the Aggset1 setting of the Aggregation Sets section in the tq-ini element is used. (See 2.2.1 for a description of aggregation sets.)

-f specifies an IOTRACE log file to be processed. *IOTRACE-log* is the name of the log file. *n* is the file cycle number. When processing multiple IOTRACE log files, multiple -f options are needed, one for each cycle to be processed. All cycles must be in sequential order and from the same IOTRACE session. You can process IOTRACE tape files that have been previously assigned to your run by specifying the file name used to assign the tape.

-s specifies the *system-name* that identifies the system source of the IOTRACE log file data. It is used to determine where to store the data in the performance database. The *system-name* parameter is case-sensitive.

Note: *Parameters passed to a program from the Exec call line are in lowercase. Some parameters on the program call statement are case-sensitive. These parameters should appear on the second and subsequent lines of the processor call.*

The parameters specified on the processor call must be separated by commas. Do not split a single parameter across two lines. No continuation character is allowed when using multiple lines. The processor call statement must be terminated with the END keyword.

The **tqiot** program processes each IOTRACE log file, storing data points in the performance database as specified by the aggregation set parameter -a and the system parameter -s.

9.8.3. Workload Characteristics for the IOTRACE Probe

The IOTRACE probe has a unique set of characteristics that are used for its workload definitions. You use these characteristics when you create the workload definitions using TeamQuest Performance Software Administration (see 5.3). The following characteristics define the IOTRACE workloads:

Characteristic Name	Valid Characters	Type and Max Length	Valid Operations
Run_ID	A–Z, 0–9	string, 6	==, !=
Device	A–Z, 0–9	string, 6	==, !=
File	A–Z, 0–9	string, 12	==, !=
TIP_File	0–9	string, 6	==, !=
Program	A–Z, 0–9	string, 3	==, !=
Transaction	A–Z, 0–9	string,12	==, !=

Example 1

The following example shows a workload definition, using run identifiers to identify system jobs. This workload contains information for system background runs MSCP, RUNNER, CMR, PCCTI, and DDP. By using the Run_ID="MSCP"||Run_ID=/MSCP./ construct, data for runs with identifiers of MSCP and MSCP*x*, where *x* can be any single alphanumeric character, is included in this workload.

```
Run_ID="MSCP"||   Run_ID=/MSCP./||   Run_ID=/RUNNE./||
Run_ID="CMR"||    Run_ID=/CMR./||
Run_ID="PCCTI"||  Run_ID=/PCCTI./||
Run_ID="DDP"||    Run_ID=/DDP./||
```

Example 2

The following example shows a workload definition using file name identifiers to group all I/O activity to a set of specific files.

```
File=={/ONL.*/,/XYZ.*/,"DIAG$","RLIB$"}
```

9.9. MAPPER Log Data Collection

Note: *The MAPPER log probe is released with TeamQuest Baseline only. It is not available with TeamQuest Online.*

The MAPPER log probe uses data from MAPPER accounting log files. The MAPPER software automatically creates this accounting log file. At various stages in the life of a MAPPER run or manual function, entries are made into this log. The entry includes information about the resources used by the function. The site can cause the MAPPER software to create additional log entries for diagnostic purposes. This is known as turning on run function logging. With run function logging off, an entry is made for each manual function, for each run and additional entries for displays, and for certain other functions. With run function logging on, an entry is made for each manual function, for each run, and for each function within a run for which there is a corresponding manual function. Run function logging can be turned on globally for all of the runs in a MAPPER system or locally for certain runs. It is recommended that sites use the local run function logging approach if this additional information is desired.

9.9.1. Data Collection Runstream

The **tqinit** process at installation time creates runstreams tailored to the MAPPER environments on your system. For every MAPPER environment to be probed, a runstream named **tq*x*mr** is created. Each of these runstreams sample a particular MAPPER environment and each must be started separately to collect MAPPER performance information. Each runstream uses a **tq*x*mr** run identifier and executes the **tqmpr** program. The **tqmpr** program processes data from a MAPPER log and stores it in the performance database.

The **tqmpr** program processes all cycles of the MAPPER log file from the previous day. During the **tqinit** process at installation time, you are prompted for the time that the **tq*x*mr** runstream starts each day. The **tq*x*mr** runstream should run each day to ensure that your performance database contains a complete set of data for each day.

9.9.2. tqmpr Processor Call Statement

The **tq*x*mr** runstreams contain the **tqmpr** processor call statement, which controls the MAPPER accounting log data reduction.

The **tqmpr** processor call statement specifies data collection parameters including input parameter table name, database name, aggregation set name, and MAPPER environment name. Each **tq*x*mr** runstream contains the **tqmpr** processor call parameters. Normally, you use the default values for these parameters as provided in the **tq*x*mr** runstream. To modify any of the defaults, you need to update the **tq*x*mr** runstream in SYSLIB*RUN$.

The **tqmpr** processor call has the following format:

```
@tqmpr [-z,] [-c,] -i,parameter-table,
       [-d,database-name,]
       [-a,aggset1,]
       -p,mapper-name,
       END
```

where:

-z is used to register the **tqmpr** probe in the database.

-c allows **tqmpr** to process MAPPER accounting files that contain Biller 1100 formatted entries.

-i specifies the location of the input parameter table name to be used by the **tqmpr** probe. For the **tqmpr** probe, the *parameter-table* is the element SYSLIB*TQ-BASELINE.MPRINPUTTBL. This is a required parameter on the **tqmpr** call line.

-d specifies the name of the database to be used for data collection. The *database-name* parameter is the name of an existing database on your system. If this parameter is not specified, the Database setting in the tq-ini element is used. Make sure that all programs called in a runstream use the same database name.

-a specifies the aggregation set to be used for data collection. The *aggset1* parameter is the name of an aggregation set defined in the *database-name*. The *aggset1* parameter is case-sensitive.

The aggregation set is optional on the call line. If no aggregation set is specified, the Aggset1 setting of the Aggregation Sets section in the tq-ini element is used. (See 2.2.1 for a description of aggregation sets.)

-p specifies a MAPPER name. The *mapper-name* parameter is the name that is associated with the MAPPER environment that generated the accounting log file. It is used to determine where to store the data in the performance database. The *mapper-name* is case-sensitive.

Note: *Parameters passed to a program from the Exec call line are in lowercase. Some parameters on the program call statement are case-sensitive. These parameters should appear on the second and subsequent lines of the processor call.*

The parameters specified on the processor call must be separated by commas. Do not split a single parameter across two lines. No continuation character is allowed when using multiple lines. The processor call statement must be terminated with the END keyword.

The **tqmpr** program processes each MAPPER accounting log file, storing data points in the performance database as specified by the aggregation set parameter -a.

9.9.3. MAPPER Log File Input

With the **tqmpr** program, you can load data from a MAPPER log file into a performance database by specifying the specific MAPPER log file. This enables you to load MAPPER log data from another system or to update the performance database with MAPPER log data that was collected at a time when **tqmpr** was not running on your system.

To load data directly from specific MAPPER log files, you must specify two additional parameters on the **tqmpr** processor call statement. (See 9.9.2 for the format of the processor call statement.) The format for these parameters is as follows:

```
-f,accntg-log(n).,[-f,accntg-log(n+1)., ...,]
-s,system-name,
```

where:

-f specifies a MAPPER accounting log file to be processed. *accntg-log* is the name of the file. *n* is the file cycle number. When processing multiple MAPPER accounting log files, multiple -f options are needed, one for each cycle to be processed. All cycles must be in sequential order.

-s specifies the *system-name* that identifies the system source of the MAPPER accounting log file data. It is used to determine where to store the data in the performance database. The *system-name* parameter is case-sensitive.

The **tqmpr** -d, -a, and -p parameters (see 9.9.2) are also used for MAPPER accounting log file input. Only one aggregation set (-a) can be specified when using the -f parameter. The -a and -p parameters are required on the **tqmpr** processor call. If you do not want **tqmpr** to load the MAPPER data into the version of the performance database specified on the Database setting in the tq-ini file, use the -d parameter to specify an alternate database version.

Note: *Before loading data into an alternate version of the performance database, you must use the* ***tqdbu*** *utility to create this alternate version (see 10.4).*

The **tqmpr** program processes each MAPPER log file, storing data points in the performance database as specified by the aggregation set parameter -a.

9.9.4. Changing Default MAPPER Log File Names

During the **tqinit** process at installation time, you are asked to provide the file name for your MAPPER log files. This information is stored in the performance database and used by **tqmpr** for selecting the MAPPER log files for processing. If you specify a cycle number, **tqmpr** selects all cycles of the MAPPER log file greater than the specified cycle. If you do not specify a cycle number, **tqmpr** selects all cycles found on the system. The cycle number of the last file processed is then written to the database. This allows **tqmpr** to process all current cycles of the MAPPER log file automatically.

To change the file name, you must specify one additional parameter (-j) on the **tqmpr** processor call statement. (See 9.9.2 for the format of the processor call statement.) The **tqmpr** program stores the new file name and its cycle number in the performance database. The **tqmpr** program uses the new information for selecting files for processing. The format of this additional parameter is as follows:

```
-j,acctng-log(n).,
```

where:

-j specifies the latest cycle of the MAPPER accounting log file name that has been processed. *acctng-log* is the name of the MAPPER accounting log file. *n* is the file cycle number. **tqmpr** will process all the cycles currently found on the system starting with cycle *n*+1.

Note: *The -j parameter and its parameter values should only be specified when new default file names are required. The parameter should be removed for normal execution of the probe. It should not be used with the -f parameter (see 9.9.3).*

9.9.5. Workload Characteristics for the MAPPER Log Probe

The MAPPER log probe has a unique set of characteristics that are used for its workload definitions. You use these characteristics when you create the workload definitions using TeamQuest Performance Software Administration (see 5.3). The following characteristics define the MAPPER log workloads:

Characteristic Name	Valid Characters	Type and Max Length	Valid Operations
User	A–Z, 0–9, ~, ^, %	string, 12	=, !=
Dept	A–Z, 0–9, ~, ^, %	string, 11	=, !=
Function	A–Z, 0–9, ~, ^, %	string, 12	=, !=
Run	A–Z, 0–9, ~, ^, %	string, 12	=, !=
Type	0–9	integer, 6	=, <, >, <=, >=, !=
Station	0–9	integer, 12	=, <, >, <=, >=, !=

Example

The following example shows a workload specification, using MAPPER department names to identify "Overhead" interactions. This workload will contain information for department names PAYROLL, and PAYROLLx, MIS, and SYSTEM. By using the Dept="PAYROLL" || Dept=/PAYROLL./ construct, data for interactions with department identifiers of PAYROLL and PAYROLL*x*, where *x* can be any single alphanumeric character, will be included in this workload.

```
Dept="PAYROLL" || Dept=/PAYROLL./ ||
Dept="MIS" || Dept="SYSTEM"
```

9.10. System Log Data Collection

Note: *The system log probe is released with TeamQuest Baseline only. It is not available with TeamQuest Online.*

TeamQuest Baseline contains a system log data collection runstream called **tqsys**. The **tqsys** runstream uses a run identifier of **tqsys** and executes the **tqsys** program. The **tqsys** program collects system log data and stores it in the performance database.

The **tqsys** program processes all cycles of the system log file from the previous day. During the **tqinit** process at installation time, you are prompted for the time that the **tqsys** runstream starts each day. The **tqsys** program should run each day to ensure that your performance database contains a complete set of data for each day.

9.10.1. tqsys Processor Call Statement

The **tqsys** runstream contains the **tqsys** processor call statement, which controls the system log data reduction.

The **tqsys** processor call statement specifies data collection parameters including input parameter table name, database name (optional), and aggregation set name. Normally, you use the default values for these parameters as provided in the **tqsys** runstream. To modify any of the defaults, you need to update the **tqsys** runstream in SYSLIB*RUN$.

The **tqsys** processor call has the following format:

```
@tqsys [-z,] -i,parameter-table,
       [-d,database-name,]
       [-a,aggset1,]
       END
```

where:

-z is used to register the **tqsys** probe in the database.

-i specifies the location of the input parameter table name to be used by the **tqsys** probe. For the **tqsys** probe, the *parameter-table* is the element SYSLIB*TQ-BASELINE.SYSINPUTTBL. This is a required parameter on the **tqsys** call line.

-d specifies the name of the database to be used for data collection. The *database-name* parameter is the name of an existing database on your system. If this parameter is not specified, the Database setting in the tq-ini element is used. Make sure that all programs called in a runstream use the same database name.

-a specifies the aggregation set to be used for data collection. The *aggset1* parameter is the name of an aggregation set defined in the *database-name*. The *aggset1* parameter is case-sensitive.

The aggregation set is optional on the call line. If no aggregation set is specified, the Aggset1 setting of the Aggregation Sets section in the tq-ini element is used. (See 2.2.1 for a description of aggregation sets.)

Note: *Parameters passed to a program from the Exec call line are in lowercase. Some parameters on the program call statement are case-sensitive. These parameters should appear on the second and subsequent lines of the processor call.*

The parameters specified on the processor call must be separated by commas. Do not split a single parameter across two lines. No continuation character is allowed when using multiple lines. The processor call statement must be terminated with the END keyword.

9.10.2. System Log File Input

With the **tqsys** program, you can load data from a system log file into a performance database by specifying the date range or the specific system log file. This enables you to load system log data from another system or to update the performance database with system log data that was collected at a time when **tqsys** was not running on your system.

When loading data using a date range, **tqsys** searches the system and processes all system log files that contain information in the date range. You must specify two additional parameters on the **tqsys** processor call statement. (See 9.3.1 for the format of the processor call statement.) The format for the date range parameters is the following:

```
-b,mmddyy[yy],
-e,mmddyy[yy],
```

where:

-b specifies the month, day, and year of the beginning date used in the log file selection.

-e specifies the month, day, and year of the ending date used in the log file selection.

If a 2-digit year is specified, the digits 00 through 27 are interpreted as the years 2000 through 2027 and the digits 28 through 99 are interpreted as the years 1928 through 1999.

To load data directly from specific system log files, you must specify two additional parameters on the **tqsys** processor call statement. (See 9.3.1 for the format of the processor call statement.) The format for these parameters is as follows:

```
-f,systemlog(n).,[-f,systemlog(n+1)., ...,]
-s,system-name,
```

where:

-f specifies the names of the system log files to be processed. *systemlog* is the name of the system log file. *n* is the file cycle number. When processing multiple system log files, multiple -f options are needed, one for each cycle to be processed. All files must be in sequential order.

-s specifies a system name. The *system-name* parameter specifies the name of the system source of the system log files. It determines where to store the data in the performance database. The *system-name* parameter is case-sensitive.

The **tqsys** -d and -a parameters are also required for system log file input (see 9.3.1). Only one aggregation set (-a) can be specified when using the -f parameter. If you do not want **tqsys** to load the system log data into the version of the performance database specified on the Database setting in the tq-ini file, use the -d parameter to specify an alternate database version.

Note: *Before loading data into an alternate version of the performance database, you must use the* ***tqdbu*** *utility to create this alternate version (see 10.4).*

If the -b and -e parameters are used with the -f parameter, **tqsys** processes the information in the specified log files for the date range specified on the -b and -e parameters.

The **tqsys** program processes each system log file, storing data points in the performance database as specified by the aggregation set parameter -a.

You can also set a relative time option when selecting the system log entries to process. The format of the relative time option is

```
-t,relative-hour
```

where:

-t specifies a relative hour when selecting system log entries to process. The valid range is 1 to 24. For example, -t,8 selects and processes the last 8 hours of system log data relative to when the **tqsys** program is started.

9.10.3. Workload Characteristics for the System Log Probe

The system log probe has a unique set of characteristics that are used for its workload definitions. You use these characteristics when you create the workload definitions using TeamQuest Performance Software Administration (see 5.3). The following characteristics define the system log workloads:

Characteristic Name	Valid Characters	Type and Max Length	Valid Operations
Run_ID	A–Z, 0–9	string, 6	=, !=
Site_ID	A–Z, 0–9	string,8	=, !=
Program	A–Z, 0–9, -, $	string, 12	=, !=
Transaction	A–Z, 0–9	string, 12	=, !=
PID	0–9	integer,12	=, <, >, <=, >=, !=
Account	A–Z, 0–9, -, period (.)	string, 12	=, !=
User_ID	A–Z, 0–9, -, period (.)	string,12	=, !=
Project	A–Z, 0–9, -, $	string,12	=, !=
Trans_Code	A–Z, 0–9, -, $	string,6	=, !=
Lib_Bank	0–9, -	string,12	=, !=

Example 1

The following example shows a workload definition, using run identifiers to identify system jobs. This workload contains information for system background runs MSCP, RUNNER, CMR, PCCTI, and DDP. By using the Run_ID="MSCP"||Run_ID=/MSCP./ construct, data for runs with identifiers of MSCP and MSCP*x*, where *x* can be any single alphanumeric character, is included in this workload.

```
Run_ID="MSCP"||   Run_ID=/MSCP./||   Run_ID=/RUNNE./||
Run_ID="CMR"||    Run_ID=/CMR./||
Run_ID="PCCTI"||  Run_ID=/PCCTI./||
Run_ID="DDP"||    Run_ID=/DDP./||
```

Example 2

The following example shows a workload definition, using run identifiers and account numbers to identify the workload. This workload contains information for all runs with identifiers that start with TQ and use account numbers in the range of 123100-123199. The workload also contains information for run TQSYS. By using the Run_ID=/TQ.*/ construct, runs with identifiers for TQ and TQ*x*, where *x* can be any string of alphanumeric characters, are considered for inclusion in the workload. By using the Account=/1231../construct, runs with account numbers of 1231*xx*, where *xx* must be a string of two alphanumeric characters, are considered for inclusion in the workload. By using the && operator, only runs that meet both conditions are included in the workload.

```
(Run_ID=/TQ.*/ && Account=/1231../) || Run_ID="TQSYS"
```

Example 3

The following example shows a workload definition using account numbers to identify system jobs. This workload contains information for all runs that use account numbers in the range of 100030 through 100039, 100120 through 100129, 100150 through 100159, or 100179.

```
Account=/10003[0-9]/ || Account=/10012[0-9]/ ||
Account=/10015[0-9]/ || Account="100179"
```

9.11. User Data Probes

Note: *The user data probes capability is available with TeamQuest Baseline only.*

TeamQuest Baseline provides the capability to create simple user probes to collect and store data into the performance database. This capability is useful if there are statistics that you would like to gather, but that are not collected by any of the system or application probes supplied with TeamQuest Baseline.

To make a simple user probe of your own, write an addstream that executes the various programs that it takes to create the input needed for the TeamQuest Baseline user probe programs. This may be SSG skeletons, Interactive Processing Facility (IPF) procedures, or a high-level language program to collect the desired statistics. You then use the TeamQuest Baseline user probe programs to read the data and store that data in the performance database. The addstream must provide the data in the format expected by the TeamQuest Baseline user probe programs.

The user probe programs that you can use when writing your own simple probe are called **tqusrprb** and **tqtblprb**. The one you use depends on the type of data you are collecting:

- Use the **tqusrprb** program when collecting performance data (see 9.11.1).

  The **tqusrprb** program stores data into the specified aggregation sets of the performance database. The data stored in an aggregation set consists of a real number value, its weight, and its sum of squares for each statistic. Examples of such statistics are CPU usage, disk space usage, network utilizations, and so on. The data from aggregation sets can be charted and analyzed using TeamQuest View, in the same way that data from **tqbsp** is charted and analyzed.

- Use the **tqtblprb** program when collecting table data (see 9.11.2).

  The **tqtblprb** program stores data into tables in the performance database, rather than into aggregation sets. You will want to use table storage for such data as process data, where there are varying numbers of data items at each interval with nonnumeric data (command name, user name, group name, and so on) or alarm data, where the data is event-based rather than time-based. You can analyze the data using TeamQuest View. This data can only be viewed as tables; it cannot be viewed as charts.

Note: *When you are experimenting with user probes, do not work on your primary database as you may create several experimental statistics that can cause your aggregation sets to expand unnecessarily. You should create an additional database using the* ***tqdbu*** *program (see 10.4) and experiment with your new user probes on a test database first. Only when you are sure of the user probe should you use it to store data in the primary database. You can then delete your test databases using* ***tqdbu****.*

9.11.1. Writing a User Probe for Performance Data

The **tqusrprb** probe uses the information from two user-created files and the parameters on its processor call to store user statistics into the performance database. The two files consist of a control file and a file that contains the data to be stored in the performance database. Parameters on the **tqusrprb** processor call line determine where the data is stored in the selected database.

There are two methods of calling the **tqusrprb** probe; calling **tqusrprb** probe each time you want to store a data sample or running the **tqusrprb** probe continuously processing multiple file cycles of PRBDATA as they become available. The characteristics and setup of the TQARG and PRBDATA files apply to both methods of calling the **tqusrprb** probe and are described below. See Section 9.11.1.2 for information on running the **tqusrprb** probe in continuous mode.

To write your own probe for performance data, you need to create the control file (TQARG.), and write an addstream or a program in the programming language of your choice to collect your statistic values and to create the data file (PRBDATA.). The addstream or program must generate ASCII (plain text) output that matches the format of the input as expected by the **tqusrprb** probe. The **tqusrprb** probe reads the statistic names and aggregation weights as input from the TQARG file. It then reads the data values as input from the PRBDATA file. Using the **tqusrprb** probe processor call options and the input, the **tqusrprb** probe creates parameters and stores values corresponding to those parameters in the performance database. By default, the **tqusrprb** probe reads the tq-ini file to determine the database and aggregation sets in which to store data.

Processor Call Format

The processor call format of **tqusrprb** is

```
@sys$lib$*tq-baseline.tqusrprb [-p,]
       [-d,databasename,] [-s,system,]
       [-g,catgroup,] [-c,category,] [-b,subcat,]
       [-a,aggset1,] [-h,]
       END
```

where:

-p	specifies that the timestamps supplied in the PRBDATA file represent "point-in-time" timestamps and data is stored within the corresponding aggset interval. Without the -p option, data is stored one interval earlier to match data collected by other probes, such as **tqbsp**.
-d,*databasename*	specifies the database into which data is stored. The *databasename* parameter is the name of an existing database on your system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).
-s,*system*	specifies the system key name in the parameter hierarchy of the statistic being stored. If the system is not specified, the name of the current system as returned in the program's initial register set is used.
-g,*catgroup*	specifies the category group key name in the parameter hierarchy of the statistic being stored. If the category group is not specified, 'User Defined' is used for the category group.
-c,*category*	specifies the category key name in the parameter hierarchy of the statistic being stored. If the category is not specified, (null) is used for category.
-b,*subcat*	specifies the subcategory key name in the parameter hierarchy of the statistic being stored. If the subcategory is not specified, (null) is used for subcategory.

-a,*aggset1* specifies the aggregation set to use for storing data. The *aggset1* parameter is the name of an aggregation set defined in the database. An aggregation set is strictly optional on the command line. If no aggregation set is specified, the Aggset1 setting of the Aggregation Sets section in the tq-ini element is used.

-h prints usage information containing command line syntax and exits.

END required keyword which ends command line option scanning.

TQARG File Input Format

The TQARG file consists of a line of statistics followed by a line of keywords describing the statistic's weights. The following example shows the required format of the TQARG file used by the **tqusrprb** program.

```
DATE   TIME  total  used  available  "%used" RESOURCE                A
NONE   NONE  LAST   LAST  LAST       LAST     NONE                   B
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A The first line of input must contain the field names for the statistics to be stored. The field names can appear in any order and must be separated by at least one blank space. The maximum length of this line is 255 characters.

The following field names are allowed:

DATE When a field is designated DATE, the value of the corresponding field in the data sample line (see 9.11.1.1) is used for the date stored. The date is in the format *mm/dd/yy*. If none of the fields in the first line of input is designated as a DATE field, the current date is used for all data samples stored. When you supply a DATE field, you must also supply a TIME field.

TIME When a field is designated TIME, the value of the corresponding field in the data sample line (see 9.11.1.1) is used for the time stored. The time is in the format *hh:mm:ss*. If none of the fields in the first line of input is designated as a TIME field, the current time is used for all data samples stored.

statistic-name The statistic-name is the name of any statistic you want to collect on your system. You can enter any number of statistic names. Statistic names containing spaces must be enclosed in double quotation marks (" ").

Some reserved words cannot be used as a *statistic-name* because they have special meaning when specified on the first line. Do not use NONE, DATE, TIME, and RESOURCE as a statistic name.

RESOURCE — When a field is designated RESOURCE, the value of the corresponding field in the data sample line (see 9.11.1.1) is used to fill the resource component of the parameter hierarchy. If no field is designated as a RESOURCE field, the resource component of the parameter hierarchy is NULL.

NONE — When a field is designated as NONE, the value for the corresponding field is ignored. The use of NONE is helpful when you want to skip an unwanted field produced by some commands.

B The second line of the input specifies the weights for the values to be stored. These weights tell the aggregator utility how to consolidate a set of data points from a finer-granularity aggregation set into a single point in a coarser-granularity aggregation set. When you specify a weight, you need to think about how you want to derive a single value from multiple samples. The fields must be space or tab delimited.

You must specify a weight for each field in the first line of the input (see A). The aggregation weights must be one of the following:

SUM — The coarser point is the sum of samples from the finer aggregation set.

AVG — The coarser point is the average of samples from the finer aggregation set.

MAX — The coarser point is the maximum of the finer values.

MIN — The coarser point is the minimum of the finer values

LAST — The coarser point is the last value chronologically from the sequence of the finer values.

FIRST — The coarser point is the first value chronologically from the sequence of the finer values.

statistic-name — The coarser point is a weighted sum of the finer points and the value of the statistic name is used as the weight for computing the weighted sum.

Given a sequence of values and weights,

$v_1, v_2, v_3, \ldots v_n$ and $w_1, w_2, w_3, \ldots w_n$,

the weighted sum is computed as

$(v_1*w_1 + v_2*w_2 + v_3*w_3 + \ldots + v_n*w_n) /$
$(w_1 + w_2 + w_3 + \ldots + w_n).$

The value of the *statistic-name* is used as the weight for the corresponding field.

NONE — For reserved fields such as DATE, TIME, NONE and RESOURCE, you must specify a weight of NONE. This is used as a place holder to record something in the weights line that corresponds to the reserved word fields.

PRBDATA File Input Format

The data to be stored into the performance database is found in the file with an internal use name of PRBDATA. This file must correspond in format to that described by the first line of the TQARG file. This file is created by a user-written process. This process can be an addstream that executes the various programs needed to create the output in the needed format. The following is an example of a PRBDATA file. This PRBDATA file could be used with the TQARG file described previously.

```
01/14/11    16:31:00     36855      11905      21270    36    area1
01/14/11    16:31:00     249991     206490     18511    92    area2
01/14/11    16:31:00     125247     19166      93561    17    area3
01/14/11    16:31:00     134839     104380     16979    86    area4
```

The lines of the PRBDATA file must contain the actual data samples that the **tqusrprb** program will store. Each sample of data consists of one or more lines. According to the command line options and the input, the **tqusrprb** program creates parameters and stores values corresponding to those parameters in the performance database. The fields must be space delimited. Any warnings and errors are written to the run's print file.

Example 1

The following example shows how to store DMS AREA overflow record statistics in the performance database. None of the TeamQuest Baseline probes collect this information and you may find it useful to have that information in your database. The DMS DMU verify command provides this information. To get this information into the performance database, you can write a runstream and execute it to provide input for the **tqusrprb** program.

The DMS DMU verify command outputs information on the areas of a DMS database. Database administrators are concerned as to when the overflow records are being filled. This can signal the time for a database area expansion. Statistics for the overflow records is part of the output from the DMU verify command. This output can be captured into a file, manipulated by an SSG skeleton, and input into the performance database using **tqusrprb**.

Part of the output from the DMU verify command is as follows:

```
.
.
.
STATISTICS FOR AREA PRIMUS$$D01  CODE 301

TOTAL STATISTICS FOR AREA
            # PAGES      TOTAL      USED    UNUSED    STD.  % UNUSED   TOTAL
                         WORDS  PER PAGE  PER PAGE  DEV.
RECORDS
PRIME          1000    1738338       340      1397     323        80   12044
OVERFLOW       3455    6140518        38      1738     157        97    2529
TOTAL          4455    7878856       106      1661     250        93   14573
.
.
.
```

An SSG skeleton can be written that inputs the DMU output as SGSs. The skeleton can then loop through the OVERFLOW SGSs and output the information in a format that **tqusrprb** can read. This skeleton creates lines of data that are in the format described by the TQARG file. For this example, the TQARG file is as follows:

```
RESOURCE "# pages" "Total words" NONE NONE NONE "% Unused" "Total Records"
NONE     AVG       AVG           NONE NONE NONE AVG        AVG
```

The SSG skeleton creates a PRBDATA file. This file is read by **tqusrprb** and the statistics are inserted into the performance database. The SSG skeleton is as follows:

```
*. output the OVERFLOW information.
*BRKPT prbdata.
*SET x = 1
*INCREMENT i to [OVERFLOW]
*   IF [OVERFLOW,i] < 7
[STATISTICS,[*x],3,1] 0 0 0 0 0 0 0
*   ELSE
[STATISTICS,[*x],3,1] [OVERFLOW,i,1,1,6,99,99]
*   ENDIF
*   SET x = [*x] + 2
*LOOP
```

The resultant PRBDATA is as follows:

```
PRIMUS$$D01  3455    6140518        38      1738     157        97    2529
```

A runstream is created that assigns the TQARG and PRBDATA files. Then the SSG skeleton is called with the breakpointed output of DMU as input SGSs. The **tqusrprb** program is then executed to insert the data into the performance database.

```
>@RUN TQUSR,accnt,TQDATA$
>@ASG,A TQARG.
>@ASG,T PRBDATA.
>@SSG,A DMUskel,DMUbrkpt.
>@EOF
>@SYS$LIB$*TQ-BASELINE.tqusrprb
>-a,10-minute,-c,Database,-b,Overflow,end
>@FIN
```

The **tqusrprb** program will create the following set of parameters:

```
system:User Defined:Database:Overflow:# pages:PRIMUS$$D01:::
system:User Defined:Database:Overflow:Total words:PRIMUS$$D01:::
system:User Defined:Database:Overflow:% Unused:PRIMUS$$D01:::
system:User Defined:Database:Overflow:Total Records:PRIMUS$$D01:::
```

The **tqusrprb** program determines the parameter hierarchy based on the options of the **tqusrprb** processor call and the statistic names from the TQARG file. Here the system is the host system id. Category group defaults to User Defined. Category is taken from the -c option and is Database. Subcategory is taken from the -b option and is Overflow. The statistic names are taken from the first line of the TQARG file. Each time this runstream is executed, **tqusrprb** will store the parameter values into the 10-minute aggregation set at the current system time.

The DATE and TIME fields provide the timestamps. If these fields are not provided, **tqusrprb** will use current date and time as the timestamp.

9.11.1.1. Data Storage

With all TeamQuest Baseline probes that store data into aggregation sets, a data sample stored at time *x* represents the data for the interval starting at time *x* rather than the interval ending at time *x*. The DATE and TIME given as input to **tqusrprb** are considered as the ending times of the interval and **tqusrprb** stores the data with a timestamp that is one interval before the given DATE and TIME. This is done to be consistent with other TeamQuest Baseline probes so that the user probe data correlates correctly with the data collected by the other probes.

For example, if some data is fed as input to **tqusrprb** as follows:

```
01/14/11     16:31:00     36855      11905     21270     36     area1
01/14/11     16:31:00    249991     206490     18511     92     area2
01/14/11     16:31:00    125247      19166     93561     17     area3
01/14/11     16:31:00    134839     104380     16979     86     area4
```

It will be stored into the 16:30:00 point reported by the 10-minute aggregation set.

If you prefer to store data within the same interval of the DATE and TIME you provide, you may specify the -p option in the **tqusrprb** call statement.

Note: *The **tqusrprb** program stores the last group of samples within any aggregation set interval being used. For example, if the PRBDATA file contains 30-second samples and you are storing into a 1-minute aggregation set, only the last 30-second sample within the 1-minute aggregation set interval is stored. In this example, your program that provides the samples in the PRBDATA file must do the combining of the 30-second samples into 1-minute samples, and then write the entire 1-minute sample to PRBDATA file so the **tqusrprb** stores the entire 1-minute data values.*

9.11.1.2. Running tqusrprb in Continuous Mode

The **tqusrprb** probe can be started in continuous probe by supplying the -m option. When running in continous mode, the **tqusrprb** program processes multiple cycles of PRBDATA as they become available, determined by the ability to exclusively assign each cycle. Your program that you supply to create the PRBDATA file must assign the next cycle by "@ASG,CP *PRBDATA* (+1).", writing data to it, then doing an "*@FREE PRBDATA*(+1).". When the **tqusrprb** probe can exclusively assign that cycle, the **tqusrprb** probe then stores the data from that cycle and either processes the next cycle or waits until the next cycle becomes available.

The **tqusrprb** probe continues to run until the console keyin TQUSR END is entered or *keyin-name* END is entered, where *keyin-name* is the name supplied through the -k option.

You may start several **tqusrprb** probes to continuously run if you have different types of data to store from different sources, such as two different formats of TQARG and PRBDATA. This can be done by placing @USE names on TQARG and PRBDATA prior to calling the **tqusrprb** probe, or by changing the qualifier on the runstream. Since keyin-names must be unique, you must supply a unique keyin-name through the -k option for runs of **tqusrprb** other than the initial run, which uses the default keyin-name of TQUSR. You stop each run of **tqusrprb** by issuing the associated END keyin-name used when **tqusrprb** was started.

The processor call format of **tqusrprb** is:

```
@sys$lib$*tq-baseline.tqusrprb [-p,]
      -m, [-n,cycle,] [-x,seconds,]
      [-r,] [-k,keyin-name]
      [-d,databasename,] [-s,system,]
      [-g,catgroup,] [-c,category,] [-b,subcat,]
      [-a,aggset1,]
      [-h,]
      END
```

where:

-p	specifies that the timestamps supplied in the PRBDATA file represent “point-in-time” timestamps and data is stored within the corresponding aggset interval. Without the -p option, data is stored one interval earlier to match data collected by other probes, such as **tqbsp**.
-m	specifies that **tqusrprb** is to run continuously.
-n,*cycle*	specifies the cycle of PRBDATA to start processing. If not supplied, the current cycle of PRBDATA is used.
-x,seconds	specifies the time (in seconds) to wait for the next cycle of PRBDATA if it is not yet availalbe. The default value is 5 seconds.
-r	specifies to run **tqusrprb** in real-time.

-k, *keyin-name*	specifies the keyin-name that is registered and used for terminating the **tqusrprb** run. The default keyin-name is TQUSR.
-d,*databasename*	specifies the database into which data is stored. The *databasename* parameter is the name of an existing database on your system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).
-s,*system*	specifies the system key name in the parameter hierarchy of the statistic being stored. If the system is not specified, the name of the current system as returned in the program's initial register set is used.
-g,*catgroup*	specifies the category group key name in the parameter hierarchy of the statistic being stored. If the category group is not specified, 'User Defined' is used for the category group.
-c,*category*	specifies the category key name in the parameter hierarchy of the statistic being stored. If the category is not specified, (null) is used for category.
-b,*subcat*	specifies the subcategory key name in the parameter hierarchy of the statistic being stored. If the subcategory is not specified, (null) is used for subcategory.
-a,*aggset1*	specifies the aggregation set to use for storing data. The *aggset1* parameter is the name of an aggregation set defined in the database. An aggregation set is strictly optional on the command line. If no aggregation set is specified, the Aggset1 setting of the Aggregation Sets section in the tq-ini element is used.
-h	prints usage information containing command line syntax and exits.
END	required keyword which ends command line option scanning.

9.11.1.3. Installing Your User Probe

After you have finished developing your user probe, you may want to edit the control stream in SYSLIB*TQ-BASELINE.TQCTL-ADD so that your probe is started along with the other probes.

9.11.1.4. Restrictions

The following restriction applies to the use of the **tqusrprb** program:

You are limited to one Category Group:Category:SubCategory in the parameter key hierarchy.

9.11.2. Writing a User Probe for Table Data

The **tqtblprb** program provides a method of creating, populating, and removing TeamQuest Baseline tables. The program is intended for use with tabular data that you would like to collect and store for analysis with the TeamQuest View user interface.

The **tqtblprb** program reads and processes ASCII input that can be produced by SSG skeletons, IPF procedures, or by programs written in any programming language.

This section explains the two-step process of creating and then populating a TeamQuest Baseline table. Throughout this section you will find references to a specific example which will provide a clearer understanding of the concepts and procedures involved in this process.

Example

At the end of each business day, you want to know how much fixed mass storage disk space is being used by each account.

Special Syntax

When creating input for **tqtblprb**, follow these general rules:

- You can use the backslash character (\) to continue input to the next line.
- Items on a line are separated by spaces or tabs.
- Items that contain spaces or special characters must be enclosed in double quotation marks (" ").
- An item enclosed in quotation marks cannot be continued to the next input line.

9.11.2.1. Creating a Table

Before tabular data can be stored into the performance database, a table must exist to hold the data. The table creation phase defines the characteristics of the table and the data to be stored in it. You must define the characteristics of the table (Table Definition) and the characteristics of the data to be stored in the table (Field Definitions). These two types of definitions comprise the Table Input Specifications. The **tqtblprb** reads the Table Input Specifications and creates the table.

Table Input Specifications

A table input specification defines a new table. Each table input specification consists of two parts: one input line that contains the table definition and one input line for each field in the table containing that field's definition.

- A table definition describes the overall table characteristics and begins with the keyword TABLE.
- Field definitions describe the characteristics of every field in the table and begin with the keyword FIELD.

Table Definition

The table definition must be the first line in the table specification and must contain the following information:

```
TABLE Table_Name=table-class:table-name \
      File_Name=file-name \
      [time-type] \
      [Retention=ndays]
```

where:

TABLE
: Defines a table input line and must be first on the line. This item is required.

Table_Name=*table-class*:*table-name*
: Specifies the table name and is a required item.

Table_Name	is the keyword for the item.
table-class	specifies the class for the table. For example, 2200 and Alarms can be classes of tables.
table-name	specifies the name for the table.

File_Name=*file-name*
: Specifies the name of the file in which to store the table and is a required item.

File_Name	is the keyword for the item.
file-name	is the file name in which to store the table.

time-type
Can be one of the following (optional). See 9.11.2.5 for a description of these time types.

Table_Time_Granulized

Table_Time_Sampled

Table_Time_Event

Retention=*ndays*
Specifies the retention period for the primary retention. A primary retention specification is automatically created when the table is created. If this optional keyword is specified, the primary retention period is set to *ndays*. Otherwise, it is set to 1 day.

Example

For the example, 2200 is the *table-class*, accountdisk is the *table-name*, and usertabs is the *file-name*. The table definition input is as follows:

```
TABLE Table_Name=2200:accountdisk File_Name=usertabs
```

The file TQDATA$**db*usertabs will be created on the system. This file should be added to the set of files that are secured on the system.

Field Definition Guidelines

Defining TeamQuest Baseline tables may appear complex when looking at the various field definition options. In reality, there are some basic guidelines that you can follow to make the process easier and that should address most requirements.

- Group Key Fields

  Each table must have one or more group keys for record sequencing and indexing. These fields should be the first fields in the table.

  In the example, Time, Interval, and System are the fields that are part of the group key. You should use these field definitions in most cases. They should be the first three fields defined.

  A unique group key normally defines one sample interval of data.

- Primary Key Fields

  Because you are storing table-oriented data, each sample interval normally has more than one record, with each record describing a unique entity that is being measured. Therefore, each table must have one or more primary keys that uniquely define the entity being measured or reported.

  Group keys are almost always primary keys, as well. However, you usually need one or more additional fields as primary keys to uniquely define each record in the table. Normally, this will be a single field that describes the entity being measured or reported such as "User" or "Account".

- Data Fields

  Data fields are the fields that normally contain the measurement data about the entity being monitored.

Field Definition

There should be one field definition for each field (column) in the table. Each field definition must be on a single input line and contain the following information.

```
FIELD Field_Name=field-name \
      [Group_Key] \
      [Primary_Key] \
      [Identifier] \
      [field-type] \
      [data-type] \
      [data-subtype] \
      [Max_Size=max-size] \
      [Time_Consolidation] \
      [Seq_Cons_Method=cons-method[,field-name]] \
      [Non_Seq_Cons_Method=cons-method[,field-name]]
```

where:

FIELD
: Defines a field input line and must be first on the line. This item is required.

Field_Name=*field-name*
: Specifies the field name and is a required item.

Field_Name	is the keyword for the item.
field-name	is the name for the field.

Group_Key
: Indicates that the field is part of the group key for the table. Normally, the sample time, sample interval, and system name make up the group key. If Group_Key is not present, the field is not part of the group key.

Primary_Key
: Indicates that the field is part of the primary key for the table. Normally, the fields that are group keys and any additional fields required to uniquely identify a record are primary keys. If Primary_Key is not present, the field is not part of the primary key.

Identifier
: Indicates that the field is an identifier for the table. An identifier describes the object being measured rather than containing a value. If Identifier is not present, the field is not an identifier field.

field-type
: Can be one of the following (optional). See 9.11.2.5 for a description of these field types.

  Field_Type_None (Default)

  Field_Type_Timestamp

  Field_Type_Desired_Interval

  Field_Type_System

data-type
: Can be one of the following (optional). See 9.11.2.5 for a description of these data types.

  U_Long_Int_Type (default)

  Long_Int_Type

  Short_Int_Type

  U_Short_Int_Type

  Float_Type

  Double_Type

  String_Type

  Byte_Array_Type

data-subtype
Can be one of the following (optional). See 9.11.2.5 for a description of these data subtypes.

No_Subtype (default)

Boolean_Subtype

Time_Subtype

Date_Subtype

Time_Date_Subtype

Elapsed_Time_Subtype

Binary_Subtype

Octal_Subtype

Hexadecimal_Subtype

Percentage_Subtype

Max_Size=*max-size*
Specifies the maximum size of the field. It is only valid for String_Type and Byte_Array_Type. If not specified, zero, which is unlimited, is used.

Max_Size	is the keyword for the item.
max-size	is the maximum size in bytes.

Time_Consolidation
Indicates the field must match when consolidating across time. If Time_Consolidation is not present, the field does not have to match.

Seq_Cons_Method=*cons-method*[,*field-name*]
Specifies a sequential consolidation method for the field. If Seq_Cons_Method is not present, it is assumed that the field cannot be consolidated.
Seq_Cons_Method is normally used when consolidating records over time.

Seq_Cons_Method	is the keyword for the item.
cons-method	can be one of the following. See 9.11.2.5 for a description of these consolidation methods.
	Id_Cons (Default)
	Min_Cons
	Max_Cons
	Sum_Cons
	Weight_Cons
	First_Cons
	Last_Cons
field-name	is valid only for Weight_Cons types and is the field to use for calculating the weight.

Non_Seq_Cons_Method=*cons-method*[,*field-name*]
Specifies a nonsequential consolidation method for the field.
Non_Seq_Cons_Method is normally used when combining records within the same time period.

Non_Seq_Cons_Method	is the keyword for the item.
cons-method	can be one of the following. See 9.11.2.5 for a description of these consolidation methods.
	Id_Cons (Default)
	Min_Cons
	Max_Cons
	Sum_Cons
	Weight_Cons
	First_Cons
	Last_Cons
field-name	is valid only for Weight_Cons types and is the field to use for calculating the weight.

Example

In the example, the first field definition is Time. Time is a group key and a primary key. The *field-type* is a timestamp and is specified as Field_Type_Timestamp. The data-subtype is time and date and is specified as Time_Date_Subtype. The field is consolidated by the last value. This field definition is as follows:

```
Field Field_Name=Time
      Group_Key \
      Primary_Key \
      Field_Type_Timestamp \
      Time_Date_Subtype  \
      Seq_Cons_Method=Last_Cons
```

Note: *This field definition is for the sample time and should be the first field in every table.*

Creating the Example Table

The following commands will create the table specification input file TPF$.dsuprobe-tab.

```
@ed,iq dsuprobe-tab
TABLE Table_Name=2200:accountdisk \                                    A
      File_Name=usertabs
FIELD Field_Name=Time \                                                B
      Group_Key \
      Primary_Key \
      Field_Type_Timestamp \
      Time_Date_Subtype  \
      Seq_Cons_Method=Last_Cons
FIELD Field_Name=Interval \
      Group_Key \
      Primary_Key \
      Field_Type_Desired_Interval \
      Elapsed_Time_Subtype \
      Seq_Cons_Method=Sum_Cons
FIELD Field_Name=System \
      Group_Key \
      Primary_Key \
      Identifier \
      Field_Type_System \
      String_Type \
      Max_Size=24 \
      Time_Consolidation \
      Seq_Cons_Method=Id_Cons
FIELD Field_Name=Account \
      Primary_Key \
      Identifier \
      String_Type \
      Max_Size=12 \
      Time_Consolidation \
      Seq_Cons_Method=Id_Cons
```

```
FIELD Field_Name="# of Fixed Files"\
      Seq_Cons_Method=Last_Cons
FIELD Field_Name="Disk Space"\
      Seq_Cons_Method=Last_Cons
@EOF
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A This line specifies the table definition. The table definition is continued on the next line by using the \ character.

B The line specifies the first field definition. This field definition is continued on the next five lines by using the \ character.

The other field definitions follow the first definition. Each definition begins with the work FIELD and starts on a separate line.

The following command takes the input file TPF$.dsuprobe-tab and creates the table:

```
@filename.tqtblprb -i, -f, dsuprobe-tab
```

This command only needs to be run once to create the table. Once the table is created, you are ready to build the runstream to perform the data collection and storage.

9.11.2.2. Storing Table Data

Data for a sample is contained on a series of input lines and is separated from other samples by a blank line. Each sample of data contains two parts: one containing the group key values and the other containing the records that describe the entities being measured in the sample interval. The group keys are contained on the first line of the sample input set. Each record is contained on a line and these lines follow the group key line. The following shows an example:

```
"01/04/12 14:00:30" 1800 CL01                                    A
123456 7 20345                                                   B
acc001 117 129678
J5KH2 246 345123
SYSTEM 520 975123
<blank line>                                                     C
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A This line specifies the Group key values for the sample. Group key fields are the fields that describe a sample interval and do not change across all of the records in that sample. This line consists of the Time, Interval, and System fields from the table specification.

In this example, the user provided the date/ time (01/04/12 14:00:30) and the system (CL01) on the input line. The value of 1800 indicates a desired sample interval of 30 minutes represented in seconds (60 seconds per minute * 30 minutes).

You can use two local variables to make date/time and system specifications easier. These variables are

$time	supplies the current date and time
$system	supplies the system name from the -s option, if specified. If the -s option is not specified, it supplies the system name from the program's initial register set.

Using these variables, the group key values become

```
$time 1800 $system
```

B This line is the first record for the sample. Record value lines contain all fields of a record for the sample interval. This line consists of the Account, # of Fixed Files, and Disk Space fields from the table specification. The value for the Account field is 123456, the value for # of Fixed Files is 7, and the value for the Disk Space field is 20345.

Additional records follow on subsequent lines.

C This line is the required blank line that separates data samples.

When creating input for data storage, the following apply:

- All field values must be separated by spaces.
- If the value itself contains spaces, it must be enclosed in double quotation marks (" ").
- Values must appear on the line in the same order as the fields are specified in the table specification.

9.11.2.3. Examples of Table Data Collection

The examples in this section show how to collect and store data into the example table.

Example 1

The commands in the following addstream collect disk space usage by account:

```
@asg,t t$pr.
@brkpt print$,t$pr.
@mfdrpt
account_summary
@eof
@brkpt print$
@ed,uq t$pr.
F   Account
+3
on dsplit
split tpf$.a
+-----
D +20
i $time 86400 $system
@eof
@filename.tqtblprb -f,t$pr.
-t,2200:accountdisk,end
```

By placing an @RUN card with a start time as the first image and an @ START card as the last image, this can be made into a runstream that can execute every day. Add an @START image to your system's AUTO$START element so the runstream will be placed in backlog.

Note: *The interval should be given in seconds. The value of 86400 is the number of seconds in 24 hours.*

Example 2

A little more sophisticated example follows. In this example, you only want to collect disk space usage by owner statistics when the amount of available fixed mass storage falls below 10 percent.

First, you need to create a threshold to check for the TeamQuest Probes **tqmsp** statistic system:MassStor by System:LOCAL::% In Use::::: to be greater than 90 percent. (See 7.4 on how to create a threshold.) Then, you need to modify the example table specification to include a *time-type* of Table_Time_Event and to remove the Interval field. Next, you need to modify the data collection runstream by removing the interval. Finally, you would add user code to USRALARM-C element that will issue an @START of the data collection runstream, compile USRALARM-C, and re-link **tqalm**. (See 7.19 on how to add user alarm code to **tqalm**.)

```
@ed,iq qual*file.dsuprobe-tab
TABLE Table_Name=2200:ownerevent\
      File_Name=usertabs \
      Table_Time_Event
FIELD Field_Name=Time \
      Group_Key \
      Primary_Key \
      Field_Type_Timestamp \
      Time_Date_Subtype \
      Seq_Cons_Method=Last_Cons
FIELD Field_Name=System \
      Group_Key \
      Primary_Key \
      Identifier \
      Field_Type_System \
      String_Type Max_\
      Max_Size=12 \
      Time_Consolidation \
      Seq_Cons_Method=Id_Cons
FIELD Field_Name=Owner \
      Primary_Key \
      Identifier \
      String_Type \
      Time_Date_Subtype \
      Time_Consolidation \
      Seq_Cons_Method=Id_Cons
FIELD Field_Name="Fixed Files" \
      Seq_Cons_Method=Last_Cons
FIELD Field_Name="Fixed Tracks" \
      Seq_Cons_Method=Last_Cons
@eof
```

The following command takes the input file qual*file.dsuprobe-tab and creates the table:

```
@sys$lib$*tq-baseline.tqtblprb -i,-f,qual*file.dsuprobe-tab
```

The following is the data collection runstream that can be placed in file qual*file.dsuprobe/run:

```
@RUN TQTBL,account,project-id
@sym,df print$
@cat,p own$prt(+1).
@brkpt print$,own$prt.
@mfdrpt
owner_summary
@eof
@brkpt print$
@ed,uq own$prt.
f      Owner
+3
on dsplit
split tpf$.a
f -------
d +222
top
i $time $system
top
tchar %
f   %
c /        /NO-OWNER/
last
i ;
@eof
@sys$lib$*tq-baseline.tqtblprb -f,own$prt.
-t,2200:ownerevent,end
@fin
```

The following is the user code updated in USRALARM-C:

```
.
.
.
/* ****************************************************************** */
/*  The process_alarm function is called each time an alarm has been  */
/*  received for processing.                                          */
/* ****************************************************************** */

#ifdef __STDC__
static void process_alarm(ALARMENTRY *alarmentry)
#else
static void process_alarm(alarmentry)
ALARMENTRY        *alarmentry;
#endif
{

#if PRINT_ALARM
   print_this_alarmentry(stdout, alarmentry);
#endif

   /*  add user-specific alarm processing code here  */

   if (strcmp(alarmentry->catgroup,"MassStor by System") == 0)
      if (strcmp(alarmentry->cat,"LOCAL") == 0)
         if (strcmp(alarmentry->subcat,"") == 0)
            if (strcmp(alarmentry->stat,"% In Use") == 0)
               if (alarmentry->current_zone != alarmentry->normal_zone)
                  facsf("@start qual*file.dsuprobe/run. ");
}
.
.
.
```

9.11.2.4. Command Syntax

The command line syntax of **tqtblprb** is

```
@filename.tqtblprb [-h,] [-i,] [-r,] [-f,file,] [-d,databasename,]
        [-t,class:name,] [-s,system,] end
```

You will use the command line options in a particular combination depending on whether you are executing **tqtblprb** for creating a table, inserting data into the table, or removing a table from the database. Each of the options are described in further detail later in this section.

If you specify the -h option, usage information containing the command line syntax is printed and the program then terminates.

Parameters passed to a program from the Exec call line are in lowercase and must have valid file name syntax. Some parameters on the **tqtblprb** call line are case-sensitive or have characters that are invalid for a file name. These parameters must appear on the second and subsequent lines of the processor call.

Command line option values should be enclosed in quotation marks if the value contains spaces.

Table Creation

The syntax for executing **tqtblprb** to create a table is

```
@filename.tqtblprb -i,[-f,file,] [-d,databasename,] end
```

where:

-i specifies a table creation execution.

-f specifies the name of a file that contains table specifications for creating the table. If the -f option is not specified, the table specifications are read from the runstream.

-d specifies the database into which data is stored. The *databasename* parameter is the name of an existing database on your system. If the *databasename* is not specified, the database in the tq.ini file is used.

The following shows the table creation command for the example. The **tqtblprb** program will read the usertabs file for creating the table.

```
@filename.tqtblprb -i,-f,usertabs.,end
```

Note: *The **tqtblprb** program sets T3 of the run's condition word to 1 to indicate the table already exists. It sets T3 to 2 for all other error terminations. This could be used in runstreams to determine whether to terminate or continue with data collection.*

You can put up to ten tables in a table file. It is a good idea to put similar tables in the same file. The TeamQuest Baseline table manager handles many operations on a file level and it is not a good idea to have a very large number of files open at one time. This example assumes that you will put all of the user-created tables in a file called usertabs. Only one table can be created with a single **tqtblprb -i** execution.

Table Storage

The syntax for executing **tqtblprb** to store data into a table is

```
@filename.tqtblprb [-f,file,] [-d,databasename,] [-s,system,]
                   -t,class:name, end
```

where:

-f specifies the name of a file that contains the data to be stored. If the -f option is specified, the data is read from the file. If the -f option is not specified, it is read from the runstream.

-d specifies the database into which data is stored. The *databasename* parameter is the name of an existing database on your system. If the *databasename* is not specified, the database in the tq.ini file is used.

-s specifies the system to use for data storage. If the -s option is present, the specified system is substituted for any occurrence of $system. If the -s option is not present, the system name found in the program's initial register set is used.

-t specifies the table class and table name for data collection and storage. This field is required.

Table Removal

The syntax for executing **tqtblprb** to remove a table is

```
@filename.tqtblprb -r, [-d,databasename,]
            -t,class:name, end
```

where:

-r specifies a table removal execution.

-d specifies the database the table is stored. The *databasename* parameter is the name of an existing database on your system. If the *databasename* is not specified, the database in the tq.ini file is used.

-t specifies the table class and table name for removal. This field is required.

> **Caution**
>
> You should take special care when removing a table. Once a table is removed, all the data stored in the table is gone. If the table was not backed up, the data stored in the table can never be recovered.

The following example shows a table removal execution that removes the 2200 accountdisk table.

```
@sys$lib$*tq-baseline.tqtblprb -r
          -t,2200:accountdisk, end
```

9.11.2.5. Keyword Definitions

Time Types Definitions

The *time-type* in a TABLE definition (see "Table Definition" in 9.11.2.1) can be one of the following:

Table_Time_Granulized	is used for cumulative data.
Table_Time_Sampled	is used for sampled data. This is the default.
Table_Time_Event	is used for event data.

Field Types Definitions

The *field-type* in a FIELD definition (see "Field Definition" in 9.11.2.1) can be one of the following:

Field_Type_None	is the type used by most fields. This is the default.
Field_Type_Timestamp	is the time of the sample.
Field_Type_Desired_Interval	is the interval of the sample.
Field_Type_System	is the system for this data.

Consolidation Methods

The *cons-method* field specifies the consolidation method for the Seq_Cons_Method and the Non_Seq_Cons_Method keywords in a FIELD definition (see "Field Definition" in 9.11.2.1). The following are allowable values for *cons-method*:

Id_Cons	Use the identifier for consolidation. Use if the identifiers are the same. Otherwise NULL the value. This is the default.
Min_Cons	Use the minimum of the values for consolidation.
Max_Cons	Use the maximum of the values for consolidation.
Sum_Cons	Use the sum of the values for consolidation.
Weight_Cons	Use the weighted summation of the values for consolidation.
First_Cons	Use the first value (time ordered) for consolidation.
Last_Cons	Use the last value (time ordered) for consolidation.

Data Types Definition

Each data value must be formatted according to the *data-type* of the field and, in some cases, according to the *data-subtype* (see "Field Definition" in 9.11.2.1).

The following apply to data value formats:

Short_Int_Type
: Subtype is not checked for this type. Allowable characters are numeric digits, the plus sign (+), and hyphen (-) if they precede the numeric digits. Any other character will cause the field (and thus the record) to be rejected. The value must be a valid short integer.

U_Short_Int_Type
: Subtype is not checked for this type. Allowable characters are numeric digits and the plus sign (+) if it precedes the numeric digits. Any other character will cause the field (and thus the record) to be rejected. The value must be a valid unsigned short integer.

Long_Int_Type
: Subtype is not checked for this type. Allowable characters are numeric digits, the plus sign (+), and hyphen (-) if they precede the numeric digits. Any other character will cause the field (and thus the record) to be rejected. The value must be a valid long integer.

U_Long_Int_Type
: Subtype is checked for the type and determines the data format rules as follow:

Time_Subtype	Time_Subtype is stored internally as seconds past midnight and the allowable input formats are *hh:mm:ss* *hh:mm* *ss*
Date_Subtype	Date_Subtype is stored internally in time_t format and the allowable input format is *mm*/*dd*/*yy*[*yy*]
Time_Date_Subtype	Time_Date subtype is stored internally in time_t format and the allowable input formats are "*mm/dd/yy*[*yy*] *hh:mm:ss*" $time
Elapsed_Time_Subtype	Elapsed_Time_Subtype is stored internally as seconds and the allowable input formats are *hh:mm:ss* *hh:mm* *ss*

All Other Subtypes	Allowable characters are numeric digits and the plus sign (+) if it precedes the numeric digits. Any other character will cause the field (and thus the record) to be rejected. The value must be a valid unsigned long integer.
Float_Type	Subtype is not checked for this type. Allowable characters are numeric digits, the plus sign (+), and hyphen (-) if they precede the numeric characters or follow the exponent character, the period (.), and the "e" and "E" characters if they follow the numeric characters and have an optionally signed value following. Any other character will cause the field (and thus the record) to be rejected. The value must be a valid floating-point value.
Double_Type	Subtype is not checked for this type. Allowable characters are numeric digits, the plus sign (+), and hyphen (-) if they precede the numeric characters or follow the exponent character, the period (.), and the "e" and "E" characters if they follow the numeric characters and have an optionally signed value following. Any other character will cause the field (and thus the record) to be rejected. The value must be a valid double-precision, floating-point value.
String_Type	Any valid characters. If the string contains spaces, it must be enclosed in double quotation marks (" ").
Byte_Array_Type	Any valid characters. If the string contains spaces, it must be enclosed in double quotation marks (" ").

Section 10
Customizing Performance Databases

10.1. Introduction

The **tqdbu** and **tqzip** programs are used to administer your performance databases. An initial database is created at installation. If you are content with the initial database, no additional administration is required. However, if you want to customize your databases, you can perform the following administrative tasks:

- Create a database (see 10.4)
- Delete a database (see 10.5)
- List databases (see 10.6)
- Estimating disk space requirements for a database (see 10.7)
- Generate a list of database parameters (see 10.8)
- Delete parameters for a database (see 10.9)
- Clearing a locked database (see 10.10)
- Backup a database (see 10.11)
- Recover a database (see 10.11)

10.2. Performance Databases

TeamQuest Baseline and TeamQuest Online use the performance database to store control information, as well as the system performance data collected on your system. Before you can use TeamQuest Baseline or TeamQuest Online, the performance database must exist on your system. The initial database is created automatically when you install TeamQuest Baseline or TeamQuest Online.

The initial performance database is named PRODUCTION. After this initial database is created, you can create additional databases to fit your particular needs. You can also delete any databases when they are no longer needed.

TeamQuest Baseline and TeamQuest Online contain a utility program called **tqdbu** that handles the creation and deletion of performance databases.

10.3. Database Files

The performance database is made up of a set of Exec files. The file TQDATA$*DBNAMES is common to all performance databases. This file is created and initialized when the first performance database is created. This file is deleted when the last database is removed.

In addition, the following files are created for each database:

- TQDATA$**xxxx*MISC
- TQDATA$**xxxx*CONFIG
- TQDATA$**xxxx*DICTION
- TQDATA$**xxxx*WORK
- TQDATA$**xxxx*THRESHLD
- TQDATA$**xxxx*LOCK
- TQDATA$**xxxxaaaaaaaa* contains the data for an aggregation set defined for the database. *aaaaaaaa* is the first eight characters of the aggregation set name. Because the aggregation set name is used in the file name, it must be unique within the first 8 characters. Also, aggregation set names can only contain the characters a–z, A–Z, 0–9, and hyphens.
- TQDATA$**xxxx*ALARMLOG is created if it does not exist when executing **tqalm**.
- TQDATA$**xxxx*ODTPFILE is created if it does not exist when executing **tqodtp**.
- TQDATA$**xxxx*SAMFILE is created if it does not exist when executing **tqsam**.
- TQDATA$**xxxx*TLIST and TQDATA$**xxxx*INTSYSTB are created if they do not exist when executing **tqalm**.

In the preceding file names, *xxxx* is the first four characters of the database name. All TeamQuest Baseline and TeamQuest Online files have a qualifier of TQDATA$.

When you are creating a database, if a database file already exists, it is initialized. If you want your performance database files to be cataloged in a particular way, you should catalog the files listed above before creating the performance database.

Note: *You should ensure that all of the files for your performance database remain loaded on your system at all times. If a database is no longer needed, be sure to use* ***tqdbu*** *to delete it.*

10.4. Creating Additional Databases

You can use the **tqdbu** utility to create additional TeamQuest databases on your system. Since a database is automatically created for you during the installation of TeamQuest Baseline or TeamQuest Online, you are not required to use **tqdbu** to create a database.

Additional databases may be desirable if you want to freeze an existing database and save it for later analysis while continuing to collect current data from your system into an alternate database. To do this, you need to use **tqdbu** to create the new database and then modify the Database setting in the tq-ini element (see 3.1). You may also want to use an alternate database to load data from a specific set of files using the TeamQuest Baseline or TeamQuest Online probes.

The **tqdbu** utility creates a database with one aggregation set in it. This aggregation set is always named 1-minute and can hold 8 hours worth of data gathered at 1-minute intervals. If you want to create additional aggregation sets for the database, you can use TeamQuest Performance Software Administration (see 6.3).

To create a database, enter the following command:

```
@tqdbu -i,database-name
```

where *database-name* is the name of the new database. Database names can be any alphanumeric name up to 12 characters in length. The names must be unique within the first four characters.

After creating a database, you may need to customize your data collection runstreams and tq-ini element to make use of the new database (see Section 9).

10.5. Deleting Databases

TeamQuest Baseline and TeamQuest Online enable you to delete a database when you no longer need it. Use the following call statement to delete a database:

```
@tqdbu -r,database-name
```

where *database-name* is the name of the database to be deleted.

After deleting a database, you must update your data collection runstreams and tq-ini element to remove references to the deleted database (see Section 9).

10.6. Listing Existing Databases

You can use the **tqdbu** utility to obtain a list of the databases that currently exist on your system. Enter the following command:

```
@tqdbu -l
```

tqdbu then displays a list of the performance databases that exist on your system.

10.7. Estimating Disk Space Requirements

If you are using TeamQuest Baseline, the amount of disk space occupied by a performance database depends on a number of different factors, mainly the number of devices available on the system.

The files that grow the most are those files containing the aggregation sets. Aggregation sets contain the data points that are to be saved for the period of time specified by the aggregation set definition. The aggregation set files have a naming convention of TQDATA$*_xxxxaaaaaaaa_* where *xxxx* is the first four characters of the database name and *aaaaaaaa* is the first eight characters of the aggregation set name. Every time an aggregation set is created, an aggregation set file is also created.

When you view an aggregation set definition with TeamQuest Performance Software Administration, the File Size field is displayed. This field can be used to estimate your sizing requirements. It contains the approximate size in tracks that the aggregation set can attain if no new parameters are added to the database. The File Size field is not shown until a probe has put at least one point of data into the aggregation set. This number can change if the probe creates additional statistics or if additional probes are used. See 6.3.3 for more information on viewing aggregation set definitions.

You can significantly reduce the disk space required to store aggregation set data by using compact aggregation sets. See 6.2.5 for a description of compact versus expanded aggregation sets.

10.8. Listing Database Parameters

You can use **tqzip** with the -o option to export a list of database parameters to a file.

Command Line Format

To generate a list of database parameters using **tqzip**, use the following command line format:

```
@tqzip [-d,databasename,] [-o,parm-file]
```

where:

-d specifies the database that **tqzip** uses. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-o outputs the database parameter list to the file specified by the *parm-file* parameter.

Parameter File Format

The -o option of the **tqzip** program generates the list of parameters and places the list in the file you specified as the *parm-file* parameter on the processor call. If you edit this file to delete parameters from the database (see 10.9), you should maintain the file in the format shown. The following shows how the list of parameters appears in the file:

```
BEGIN PARAMETER LIST created from <PRODUCTION> on Fri Jan 13 14:05:45 2012    A
System:CatGroup:Cat:SubCat:Stat:Res:App:WL Set:WL

CL01:System:::Elapsed Time::::                                                B
CL01:Processor:Processor:Utilization:Real Time:IP000:::
CL01:Processor:Processor:Utilization:Real Time:IP001:::
CL01:Processor:Processor:Utilization:Real Time:IP002:::
CL01:Processor:Processor:Utilization:Real Time:IP003:::
CL01:Processor:Processor:Utilization:Exec0:IP000:::
CL01:Processor:Processor:Utilization:Exec0:IP001:::
CL01:Processor:Processor:Utilization:Exec0:IP002:::
CL01:Processor:Processor:Utilization:Exec0:IP003:::
CL01:System:Throughput::TIP Txns/Sec::::
CL01:System:Throughput::Demand Runs Opened::::
CL01:System:Throughput::Batch Runs Opened::::
.
.
.
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A The first two lines of the file contain heading information. The second line shows the parameter hierarchy of key names. It is followed by a blank line.

B Each parameter is listed by its key name hierarchy: system, category group, category, subcategory, statistic, resource, application group, workload set, and workload.

All key values must be specified in each parameter reference.

An asterisk (*) can be used as a mask indicator instead of specifying individual key values in the parameter references. The asterisk can be used for any of the keys to simplify the deletion of sets of parameters. For example, all parameters that are for a given disk name can be deleted by specifying an asterisk in the statistic key and the disk name as the resource key, along with all of the other required key values.

10.9. Removing Database Parameters from a Database

You can remove unwanted database parameters from a database. For example, if you have removed a particular disk device from your system, you may no longer need the data collected for that device.

To remove database parameters using **tqzip**, you can generate a file containing a parameter list using the -o option. You can edit this file to contain only the parameters that you want to delete from your database. Then use the -p option on the processor call to delete the parameters specified in the parameter file.

When you delete a parameter from the database, all of the data values for that parameter are cleared in each of the aggregation sets in the database. All space previously used by deleted parameters is automatically reused when new parameters are created. Also, any threshold definitions are removed for the parameters being deleted. Disk space is not released when a parameter is deleted.

Note: *Parameter deletion should be a controlled operation and is best performed after you have terminated all TeamQuest Baseline probes executing on your system. Be careful that you do not delete any parameters that are being used by a probe that is currently executing on your system.*

Command Line Format

To remove parameters from your database using **tqzip**, use the following processor call format:

```
@tqzip [-d,databasename,] [-p,parm-file]
```

where:

-d specifies the database that **tqzip** uses. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-p deletes parameters from the database using the file specified by the *parm-file* parameter. The *parm-file* used with the -p option must be in the same format as the file produced by the -o option (see 10.8).

Example

To delete database parameters, generate a parameter list file using the -o option of **tqzip** (see 10.9). Then edit the file. The following example shows the contents of a parameter file pfile:

```
BEGIN PARAMETER LIST created from <PRODUCTION> on Fri Jan 13 14:05:45 2012
System:CatGroup:Cat:SubCat:Stat:Res:App:WL Set:WL

CL01:IO Activity:Disk:Summary:*:DSKF7:::

END PARAMETER LIST
```

This shows that you want to delete all statistics for DSKF7. You then call **tqzip** with the -p option and specify pfile on the processor call.

```
>@tqzip -p,pfile
>TQZIP 7R4G (011312 1120:33) Fri Jan 13 14:50:17 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>
>Using database TQDATA:PRODUCTION
>
>...generating list of pnums to delete
>14 parameters selected for deletion
>Proceed with deletion (y,<n>)?y
>
>...deleting parameters
>
>14 parameters were deleted.
>
>End TQZIP
```

10.10. Database Locking

The aggregation set database files (that is, those files named TQDATA$**xxxxaaaaaaaa*) are locked by each probe during an update. When one probe has an aggregation set file locked, all other probes will wait until it becomes available before storing their data. If a probe terminates abnormally, its contingency handling will ensure that any locks held by the probe are cleared. If the contingency handler does not receive control on abnormal termination, locks may not be cleared.

The **tqunlk** program is provided for clearing any locks left by abnormally terminated probes. This processor is executed in each of the probe runstreams immediately following the probe processor call statement. This execution will ensure that all locks created by the generated run-id of the probe run are cleared.

Normally, you should never need to execute **tqunlk** to clear locks. If you need to use **tqunlk**, be careful to ensure that you are not clearing a lock that is required by a probe that is currently executing on your system.

The **tqunlk** processor call has the following format:

```
@tqunlk [-c,] [-l,] [-q,qualifier,] [-d,databasename,] [-r,runid,]
        [-g,] [-z]
```

where:

-c Clear any locks currently being held. Either the -c option or the -l option must be specified. However, do not use the -c and -l options together in one execution of the **tqunlk** program.

-l List any locks currently being held. Either the -c option or the -l option must specified. However, do not use the -c and -l options together in one execution of the **tqunlk** program.

-q Use the specified database qualifier. The default qualifier is TQDATA$.

-d Use the specified database name. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-r Limit the action specified by the -c or -l option to the specified run-id. Do not use the -r and -g options together in one execution of the **tqunlk** program.

-g Limit the action specified by the -c or -l option to the generated run-id of this run. Do not use the -r and -g options together in one execution of the **tqunlk** program.

-z Use with the -l option to obtain a list of any locks currently being held for all database qualifiers, database names, and run-ids.

10.11. Backup and Recovery

You should save and restore the contents of the individual files that make up the performance database as a group. If, for some reason, you need to restore one of the database files from a backup, you should restore all of its companion files from the same backup.

Caution

Back up the performance database periodically. You might have to restore the database in the case where a TeamQuest Baseline probe or the **tqagg** program is abnormally terminated and leaves the database files in an inconsistent state.

Section 11
Customizing Tables

11.1. Introduction

The **tqzip** program is used to maintain the tables in your performance database. If you are happy with the default values used, no additional administration is required. Defaults are set by the probes when the tables are initially created. However, if you want to customize your data collection, you can perform the following administrative tasks:

- List tables (see 11.4)
- Set collection periods (see 11.5)
- Define retention specifications (see 11.6)
- Remove records from tables (see 11.7)
- Register tables in the database (see 11.8)
- Deregister tables in the database (see 11.9)
- Unload data collection and retention specifications (see 11.10)
- Load data collection and retention specifications (see 11.11)
- Reorganizing table file in the database (see 11.12)

11.2. Concepts

There are some basic concepts essential to the understanding of the data stored in TeamQuest Baseline tables. Before beginning any administrative tasks, you will want to familiarize yourself with these concepts.

11.2.1. Collections

Collections specify the time during a day that data records should be collected for a table. If the data stored in the table is collected at regular intervals, the interval between collections is also specified. The minimum interval allowed is specified by the table creator. Any number of collections can be defined for a given table. Collection periods cannot overlap each other.

11.2.2. Retentions

Retentions control how long data is retained in a table. If the records are created at regular intervals, the retention specifies how long the data should be retained at a given interval. Records are kept as they are initially stored, at the interval the data collector inserts them. This first retention period is referred to as the primary retention period. If records are not collected at regular intervals (such as alarm records), then the primary retention period is the only retention period that can be specified.

If the data is intervalized, additional retention periods can be defined. At the end of the primary retention period, the records are aggregated into records whose interval matches the interval specified for the next retention period. It is assumed that retention intervals increase from smallest for the most recent records, to largest for the oldest records. When the end of the oldest retention interval is reached, the records are removed from the table.

11.2.3. Registration

Table data is kept in files on the system where the database resides. The table list that can be displayed using **tqzip** shows the name of the file that contains the data for a given table. Table files are completely self-contained. Thus, they can be saved to a lower cost storage medium (such as tape) and reloaded later for examination. In order for TeamQuest Baseline to know about the reloaded table files, the table file must be registered with TeamQuest Baseline. If you do not save table files to tape, you will never need to register a table.

Since the name of the table being registered is likely to match the name of an existing table, a new name must be provided for the registered table. When a table file registered in this manner is no longer needed, it can be deregistered from the performance database. Deregistering a file does not cause a table file to be deleted. It simply removes the record of the table from TeamQuest Baseline.

11.2.4. Manual Record Deletion

Table records are normally deleted from tables by the TeamQuest Baseline aggregation program based on the retention specifications for the table. There may be times when you want to remove records from a table manually. This can be accomplished for a given system, for all records before a given point in time, or for all records in the table.

11.3. Opening the Table Administration Menu

The **tqzip** program allows you to perform the various tasks associated with maintaining tables in the performance database. The **tqzip** program displays a series of menus for you to select the tasks you want to perform. When you execute the **tqzip** program, it displays the Administration Menu that allows you to select the type of administration tasks you want to perform.

To perform maintenance tasks on tables, you execute the **tqzip** program and select Tables by entering **2** from the Administration Menu.

```
>@tqzip
>TQZIP 7R4G (011312 1104:45) Fri Jan 13 09:19:55 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>
>Using database TQDATA:PRODUCTION
>
>
>      Administration Menu
>
>    1)  Alarms and Thresholds
>    2)  Tables
>
>   99)  Exit TQZIP
>
>
>Enter your selection:
>2
```

The tqzip program then displays the Tables Menu.

```
>      Tables Menu
>
>    1)  List Tables
>    2)  Collection Periods...
>    3)  Retention Periods...
>    4)  Record Deletion...
>    5)  Table Registration...
>
>   99)  Exit Tables Menu
>
>Enter your selection <1> :
```

To perform a maintenance task, enter its associated number at the prompt. To return to the Administration Menu, enter **99**.

When you select the various maintenance tasks for the Tables Menu, the **tqzip** program prompts you to supply values or other information. Default values are displayed for each prompt in angle brackets (<>). To use a default, press the transmit key without entering a value.

11.4. Displaying a List of Tables

You can view a list of the tables in your database that contain performance information. To display the list, select List Tables from the Tables Menu (see 11.3) by entering **1** at the prompt.

```
>Enter your selection <1> :
>1
>
>  Class                 Name                  Rev  File
>  --------------------  --------------------  ---  ----
>  2200                  Runs                    1  tqdata$*prodsamfie
>  Alarm                 AlarmLog                2  tqdata$*prodalarmlg
>  2200                  Transactions            1  tqdata$*prodsamfie
>
>Continue...
```

The list of tables generated by **tqzip** includes the table class, table name, table revision, and the name of the file that contains the tables. It is possible for a file to contain more than one table.

11.5. Administering Collection Periods

Collection periods are used to control when data is collected for a given table. With the **tqzip** program, you can

- List the currently defined collection periods
- Add a collection period
- Specify that individual days of the week are excluded from the collection
- Specify how often the data values are obtained, if the table contains intervalized data (that is data where the values are obtained at regular intervals)
- Turn off data collection completely for a table while retaining the collection periods already defined

To begin the administration of the collection periods for tables, select Collection Periods from the Tables Menu (see 11.3) by entering **2** at the prompt. The **tqzip** program lists the tables defined in the database and asks you for the name of the table whose collections you wish to administer.

```
>Enter your selection <1> :
>2
>
> 1)  2200                  Runs
> 2)  Alarm                 Alarm Log
> 3)  2200                  Transactions
>Enter Table Number <enter by name> :
>
>Class    :
>2200
>Name     :
>Runs
```

The list of tables depends on the particular types of data being stored in tables. Enter the number of the table whose collections you wish to administer. If you do not enter anything, **tqzip** asks you for the complete table name, including the table class and table name, as shown in this example.

After you have selected a table, **tqzip** displays the following Collection Period Menu:

```
>      Collection Period Menu (2200:Runs)
>
>    1)  List Collection Periods
>    2)  Add a Collection Period...
>    3)  Delete a Collection Period...
>    4)  Turn Data Collection Off
>
>   98)  Omit Collection Period Changes
>   99)  Save Changes and Exit Collection Period Menu
>
>Enter your selection <1> :
```

To perform a collection period task, enter its associated number at the prompt. To accept any changes you may have made on subsequent collection period screens and to return to the Tables Menu, enter **99**. To omit any changes and to return to the Tables Menu, enter **98**.

11.5.1. Listing Collection Periods

To display a list of the currently defined collection periods for the table, select List Collection Periods from the Collection Period Menu by entering **1** at the prompt. The **tqzip** program displays the following list.

```
>Enter your selection <1> :
>1
>
>Data Collection is On.   Collection Periods (2) :                    A
>
>                         Exclude
>Number  Start     End     SMTWTFS  Interval                          B
>------  -----     -----   -------  --------
>        00:00     08:00            <No Collection                    C
>   1)   08:00     11:00   S     S   00:01:00
>        11:00     13:00            <No Collection>
>   2)   13:00     16:00   S     S   00:00:30
>        16:00     24:00            <No Collection>
>Continue...
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A The first line indicates that the data collection is ON and that 2 data collection periods are defined.

B The column headings for the data collection periods indicate the number of the collection, the start and end times, the days of the week that are excluded from the data collection, and the interval at which the data is collected if appropriate for the table.

C The list of data collection periods shows the currently defined collection periods, as well as periods when no data collection has been defined.

During the first collection period, from 8:00 to 11:00, data is collected at 1-minute intervals. During the second collection period, from 13:00 to 16:00, data is collected at 30-second intervals. For both collections, data is not collected on Saturday or Sunday. During the other periods of the day, no data collection takes place.

11.5.2. Adding a Collection Period

The **tqzip** program allows you to add new collection periods for a table. The following rules apply when adding a collection period:

- A collection period can start at any time of the day and end at any point within 24 hours of the collection start time.
- No collection period can overlap another collection period.
- Collection intervals must be such that there are an even number of collection intervals in a day and they must be an even multiple of the minimum collection interval defined by the table creator.

To add a collection period, select Add a Collection Period from the Collection Period Menu (see 11.5) by entering **2** at the prompt. The **tqzip** program displays the following series of prompts:

```
>Enter your selection <1> :
>2
>
>Interval (mm:ss)   :                                                    A
>1:00
>Start Time (hh:mm) :                                                    B
>8:00
>End Time (hh:mm)   :                                                    C
>11:00
>Mask for days of the week to exclude (Sun, Mon, ..., Sat)               D
>1000001
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A — Enter the interval at which data is collected for the collection period. If the table does not collect data at regular intervals (such as the Alarm Log table), then this prompt does not appear.

B — Enter the start time for the collection period.

C — Enter the end time for the collection period.

D — Enter the mask for the days of the week when data is not collected. The mask is a series of seven digits corresponding to the days of the week, beginning with Sunday and ending with Saturday. The mask is specified using the digit 1 to exclude a day, and the digit 0 to include a day. In the example, the mask 1000001 indicates that Sunday and Saturday are excluded from data collection.

If you do not enter a mask, all days of the week are included in data collection.

11.5.3. Deleting a Collection Period

To delete a collection period for a table, select Delete a Collection Period from the Collection Period Menu (see 11.5) by entering **3** at the prompt. The **tqzip** program displays the following prompts:

```
>Enter your selection <1> :
>3
>
>Collection Number to Delete (<cancel>, ?, #) :                          A
>?
>Data Collection is On.   Collection Periods (2)       :                 B
>
>                           Exclude
>Number  Start     End     SMTWTFS   Interval
>------  -----    -----  ---------  -------
>        00:00    08:00             <No Collection>
>   1)   08:00    11:00   S      S   00:01:00
>        11:00    13:00             <No Collection>
>   2)   13:00    16:00   S      S   00:00:30
>        16:00    24:00           <No Collection>
>Collection Number to Delete (<cancel>, ?, #) :                          C
>2
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A — The **tqzip** program prompts you for the number of the collection period that you want to delete.

If you do not know the number of the collection period, enter a question mark (?) to list the currently defined collection periods for the table.

B — The **tqzip** program displays the list of collection periods for the table.

C — The **tqzip** program prompts you again for the number of the collection period you want to delete. In this example, collection number 2 (13:00–16:00) is deleted.

11.5.4. Turning Data Collection On and Off

The Collection Period Menu (see 11.5) indicates if data collection is to be done for the table. If data collection is currently turned off, the menu selection allows you to turn it on by entering **4** at the prompt. If data collection is turned on, the menu selection allows you to turn it off by entering **4** at the prompt. By using this selection, you can stop collecting data for a table while retaining the collection periods currently defined. Removing all collection periods for a table would have the same effect as turning off data collection.

11.5.5. Exiting the Collection Period Menu

When you exit the Collection Period Menu, you can apply any collection period changes you made by selecting Exit Collection Period Menu. Enter **99** at the prompt.

If you do not want to apply the collection period changes, you can select Omit Collection Period Changes from the menu. Enter **98** at the prompt.

```
>      Collection Period Menu (2200:Runs)
>
>    1)  List Collection Periods
>    2)  Add a Collection Period...
>    3)  Delete a Collection Period...
>    4)  Turn Data Collection Off
>
>   98)  Omit Collection Period Changes
>   99)  Save Changes and Exit Collection Period Menu
>
>Enter your selection <1> :
>99
```

When you select **98** or **99** at the prompt, the **tqzip** program returns you to the Tables Menu (see 11.3).

11.6. Defining Retention Specifications

Retention specifications define how long data is kept for tables. The first retention specification is for data that is kept as it is stored by the data source, and is called the primary retention. For tables that contain intervalized data, additional retention specifications can be defined. These subsequent retention specifications follow the primary retention and define how long and at what interval data is retained. It is assumed that larger intervals follow smaller intervals. For example, you might define a 5-minute, 2-week retention and a 30-minute, 3-month retention. The data is kept at 5-minute intervals for two weeks and then an additional three months at 30-minute intervals. This results in a total of three months and two weeks worth of data being retained.

The data in the primary retention specification is placed in the table by data collectors. The data in subsequent retentions is created by the TeamQuest Baseline aggregator based on the originally collected data. When the end of a retention period has been reached, the Baseline aggregator removes the data from the table.

When the aggregator creates new records, it will synchronize them with even intervals for a day. You supply this retention start time when you create the retention specifications. Normally, a retention start time of midnight is adequate. Thus for 30-minute data points, you would have records created at 00:00, 00:30, 1:00, 1:30, 2:00, and so on. There are times when you may want the synchronization time to be something other than midnight. For example, you might want a single point for each shift of a work day. The shifts run from 7:00 to 15:00, 15:00 to 23:00, and 23:00 to 7:00 the next day. In this case, you would want the synchronization time to be 7:00 (though it could be specified as 15:00 or 23:00 with the same results).

The primary retention contains data at whatever interval it is collected. You can only modify the retention period and not the interval of the retained data.

If the data stored in the table is not intervalized (such as the Alarm Log), no retentions other than the primary retention can be defined.

To begin the administration of the retention specification for a table, select Retention Periods from the Tables Menu (see 11.3) by entering **3** at the prompt. The **tqzip** program lists the tables defined in the database and asks you for the name of the table whose retention specification you want to administer.

```
>Enter your selection <1> :
>3
>
> 1)  2200                  Runs
> 2)  Alarm                 Alarm Log
> 3)  2200                  Transactions
>Enter Table Number <enter by name> :
>
>Class    :
>2200
>Name     :
>Runs
```

The list of tables depends on the particular types of data being stored in tables. Enter the number of the table whose collections you wish to administer. If you do not enter anything, **tqzip** asks you for the complete table name, including the table class and table name, as shown in this example.

After you have selected a table, **tqzip** displays the following Retention Specification Menu:

```
>      Retention Specification Menu (2200:Runs)
>
>    1)  List Retention Specifications
>    2)  Modify Primary Retention Specification...
>    3)  Add a Retention Specification...
>    4)  Delete a Retention Specification...
>    5)  Set Retention Start Time...
>
>   98)  Omit Retention Specification Changes
>   99)  Save Changes and Exit Retention Specification Menu
>
>Enter your selection <1> :
```

11.6.1. Listing Retention Specifications

To display a list of the currently defined retention specifications for the table, select List Retention Specifications from the Retention Specification Menu (see 11.6) by entering **1** at the prompt. The **tqzip** program displays the following list:

```
>Enter your selection <1> :
>1
>
>Retention Start Time:                                                    A
>8:00
>
>Retention Specifications (4) :                                           B
>
>   Number    Interval    Keep                                           C
>  --------   --------  ---------
>  Primary    --------   1 day                                           D
>         1)   00:01:00   1 week
>         2)   00:30:00   1 month
>         3)   01:00:00   6 months
>Continue...
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A The first line indicates the retention start time for the table. This time is used for synchronizing data collection. In this example, the start time is 8:00.

B This indicates the number of retention specifications. In this example, there are four specifications.

C The column headings for the retention specifications indicate the number of the specification, the interval at which the data is collected, and the length of time the data is kept in the database.

D The list of retention specifications shows the current retentions.

The retentions are listed in the order in which the data is retained. The primary retention specification is always listed first, followed by the additional retention specifications.

In this example, there are four retentions. The data in the primary retention is kept for 1 day at whatever interval it is collected. Retention number 1 stores data at 1-minute intervals for 1 week. Retention number 2 stores data at 30-minute intervals for 1 month. Retention number 3 stores data at 1-hour intervals for 6 months.

In terms of storing the data, the data in the primary retention is kept for one day. At the end of the day, the aggregator aggregates the original records into 1-minute intervals and removes the records that were aggregated. The 1-minute interval data is kept for 1 week. At the end of a week, the aggregator aggregates the 1-minute records into 30-minute intervals and removes the 1-minute interval records that were aggregated. The 30-minute interval data is kept for 1 month. At the end of a month, the aggregator aggregates the 30-minute records into 1-hour intervals and removes the 30-minute interval records that were aggregated. The 1-hour records are retained for 6 months. At the end of the 6 months, the records are removed from the table. For this example, data records are retained for a total of 7 months (1 month of 30-minute interval data and 6 months of 1-hour data), 1 week (1-minute interval data), and 1 day (data at the interval collected).

11.6.2. Modifying the Primary Retention Specification

The primary retention specification specifies how long the records created by the data collector will be retained. The interval of the records remains the same as when the data collector inserted them. Only the retention period can be modified for the primary retention specification.

To modify the primary retention period specification, select Modify Primary Retention Specification from the Retention Specification Menu (see 11.6) by entering **2** at the prompt. The **tqzip** program displays the following prompt:

```
>Enter your selection <1> :
>2
>
>Retention period <cancel> :
>2 days
```

At the prompt, enter a number and a unit for the retention period. Valid units are minute, hour, day, week, month, and year.

In the example, the retention period for which the data collector will create records is changed to 2 days.

11.6.3. Adding a Retention Specification

The **tqzip** program allows you to add a retention specification to a table. When you add a retention specification, the order of the retention specifications is assumed to be in increasing interval size, from the primary retention up to the largest interval.

Note: *Tables containing nonintervalized data (such as the Alarm Log table) cannot have additional retention specifications because the data records cannot be aggregated across time.*

Select Add a Retention Specification from the Retention Specification Menus (see 11.6) by entering **3** at the prompt. The following prompts are displayed:

```
>Enter your selection <1> :
>3
>
>Minimum Interval (hh:mm:ss) :                                          A
>15:00
>Retention period (i.e. 3 weeks) :                                      B
>1 week
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A Enter the interval for the data records.

B Enter the retention period. You must enter a number and a unit. Valid units are minute, hour, day, week, month, and year.

In the example, a retention specification is added to the existing set of retentions. The interval of the retained records is 15 minutes and the data is retained for 1 week. When you add this retention to the existing set, the retentions now appear as:

```
>
>Retention Specifications (5) :
>
>   Number    Interval    Keep
>  --------   --------  ---------
>  Primary    --------   1 day
>         1)   00:01:00   1 week
>         2)   00:15:00   1 week
>         3)   00:30:00   1 month
>         4)   01:00:00   6 months
>Continue...
```

Notice that the 15-minute retention falls between the 1-minute and 30-minute retentions. The data is retained at 1-minute intervals for 1 week, then at 15-minute intervals for another week, then at 30-minute intervals for 1 month, and at 1-hour intervals for 6 months.

11.6.4. Deleting a Retention Specification

You can delete any retention specifications for a table except the primary retention. The primary retention specification must always exist. Deleting a retention specification does not result in the immediate deletion of the data in that retention period. The next time the aggregator processes the data in the table, the data records will be aggregated so that they match the new retention specifications.

To delete a retention specification for a table, select Delete a Retention Specification by entering **4** at the prompt. The **tqzip** program displays the following prompts:

```
>Enter your selection <1> :
>4
>
>Retention to Delete (<cancel>, ?, #) :                                  A
>?
>Retention Specifications (5) :                                          B
>
>   Number    Interval    Keep
>  --------   --------  ---------
>  Primary    --------    2 days
>       1)    00:01:00    1 week
>       2)    00:15:00    1 week
>       3)    00:30:00    1 month
>       4)    01:00:00    6 months
ARetention to Delete (<cancel>, ?, #) :                                  C
A1
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A — The **tqzip** program prompts you for the number of the retention that you want to delete.

If you do not know the number of the retention, enter a question mark (?) to list the currently defined retentions for the table.

B — The **tqzip** program displays the list of retentions for the table.

C — The **tqzip** program prompts you again for the number of the retention you want to delete. In this example, retention number 1 (1-minute data, retained for 1 week) is deleted.

After the deletion, data will be retained at the original collection interval for 2 days, then aggregated into 15-minute records that will be retained for 1 week.

11.6.5. Setting the Retention Start Time

The retention start time is the synchronization time for aggregated records. All aggregated records will start on even intervals of the retention start time. In most cases, the retention start time can be left at 00:00. However, the **tqzip** program allows you to change the retention start time, if needed.

To set the retention start time, select Set Retention Start Time from the Retention Specification Menu (see 11.6) by entering **5** at the prompt. The **tqzip** program then displays the following prompt:

```
>Enter your selection <1> :
>5
>
>Retention Start (hh:mm) <00:00> :
>8:00
```

Enter the new start time in *hh:mm* format. The retention start time can be any time of the day, but it must be an even multiple of the minimum collection period for the probe.

11.6.6. Exiting the Retention Specification Menu

When you exit the Retention Specification Menu, you can apply any retention specification changes you made by selecting Exit Retention Specification Menu. Enter **99** at the prompt.

If you do not want to apply the retention specification changes, you can select Omit Retention Specification Changes from the menu. Enter **98** at the prompt.

```
>
>      Retention Specification Menu (2200:Runs)
>
>    1)  List Retention Specifications
>    2)  Modify Primary Retention Specification...
>    3)  Add a Retention Specification...
>    4)  Delete a Retention Specification...
>    5)  Set Retention Start Time...
>
>   98)  Omit Retention Specification Changes
>   99)  Save Changes and Exit Retention Specification Menu
>
>Enter your selection <1> :
>99
```

When you select **98** or **99** at the prompt, the **tqzip** program returns you to the Tables Menu (see 11.3).

11.7. Deleting Records from Tables

Normally, records are deleted from tables by the TeamQuest Baseline aggregator based on the retention specifications for each table. However, there may be times when you want to remove data records manually by using the **tqzip** program.

To begin record deletion for a table, select Record Deletion from the Tables Menu (see 11.3) by entering **4** at the prompt. The **tqzip** program lists the tables defined in the database and asks you for the name of the table for which you want to delete records.

```
>Enter your selection <1> :
>4
>
> 1)  2200                  Runs
> 2)  Alarm                 Alarm Log
> 3)  2200                  Transactions
>Enter Table Number <enter by name> :
>
>Class    :
>2200
>Name     :
>Runs
```

The list of tables depends on the particular types of data being stored in tables. Enter the number of the table whose records you wish to delete. If you do not enter anything, **tqzip** asks you for the complete table name, including the table class and table name, as shown in this example.

After you have selected a table, **tqzip** displays the following Data Deletion Menu:

```
>      Data Deletion Menu (2200:Runs)
>
>    1)  Delete Records Before Time...
>    2)  Delete Records for System...
>    3)  Delete All Records
>
>   99)  Exit Data Deletion Menu
>
>Enter your selection <1> :
```

Using the selections of this menu, you can delete records based on time, based on the system, or all records in the table.

Note: *Data deletion takes place immediately. There is no opportunity to stop the deletion of records from a table. Since data deletion takes place immediately, it is recommended that you back up your database before you delete any records from a table.*

11.7.1. Deleting Records Based on Time

The **tqzip** program allows you to delete all records whose timestamp is before a specified time. Select Delete Records Before Time from the Data Deletion Menu by entering **1** at the prompt. The following prompts are displayed:

```
>Enter your selection <1> :
>1
>
>Delete all records before the following date and time...
>Enter date (mmddyyyy) :                                                A
>01132012
>Enter time (hh:mm) :                                                   B
>00:00
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A Enter the date before which all records will be deleted.

B Enter the time before which all records will be deleted.

In the example, all records whose timestamp is before midnight on January 13, 2012, will be deleted from the table.

11.7.2. Deleting Records for a System

The **tqzip** program allows you to delete all records for a given system from the table. Select Delete Records for System from the Data Deletion Menu by entering **2** at the prompt. The following prompts are displayed:

```
>Enter your selection <1> :
>2
>
>Enter system to delete :
>venus
```

At the prompt, enter the system name for which all records will be deleted. In the example, all records whose system is venus will be deleted.

11.7.3. Deleting All Records for a Table

The **tqzip** program allows you to delete all records for a table. Select Delete All Records from the Data Deletion Menu by entering **3** at the prompt. When you make this selection, the **tqzip** program prompts you one more time to verify your selection. All records will then be deleted from the table.

Note: *Data deletion takes place immediately. There is no opportunity to stop the deletion of records from a table. Since data deletion takes place immediately, it is recommended that you back up your database before you delete any records from a table.*

11.8. Registering a Table

The **tqzip** program allows you to register table files that you may have previously saved (perhaps to tape) so they can be examined with TeamQuest Baseline. Because TeamQuest Baseline allows only one table to exist with a given name, you will be required to rename the table you are registering. A table file can contain more than one table. If a file has more than one table, you must specify which table from the file you want to register. You are responsible for specifying the complete file name of the table file and ensuring the file is available for use when the table is accessed.

To register a table, select Table Registration from the Tables Menu (see 11.3) by entering **5** at the prompt. The **tqzip** program displays the following Table Registration Menu:

```
>Enter your selection <1> :
>5
>
>      Table Registration Menu
>
>    1)  Register a Table...
>    2)  Deregister a Table...
>
>   99)  Exit Table Registration Menu
```

Enter **1** at the prompt to register a table. The **tqzip** program displays the following prompts:

```
>Enter your selection <1> :
>1
>
>Table filename     :                                                    A
>savefile
>
>File 'savefile' contains tables:                                        B
>
>1)  2200:Runs
>
>Registering table 2200:Runs
>New table class    :                                                    C
>2200 Archive
>New table name     :                                                    D
>Runs
>Table 2200 Archive:Runs registered.                                     E
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A Enter the name of the file that contains the table you want to register.

B The **tqzip** program prompts you with a list of tables included in this file. Because the file in this example contains only one table, it provides a message that it is registering the table 2200:Runs.

C The **tqzip** program prompts you for a new table class.

D The **tqzip** program prompts you for a new table name.

E In this example, the table called 2200:Runs in the file savefile is registered in the TeamQuest Baseline table list under the name 2200 Archive:Runs. When selecting data from this table for reporting, you would choose the table 2200 Archive:Runs. The file name specified should be the complete name of the file. When a table is registered, the table name in the file is changed. Thus if this file is registered again, the list of tables in the file will be changed to reflect the previous registration.

11.9. Deregistering a Table

When a registered table is no longer needed, it can be deregistered from TeamQuest Baseline. When you deregister a table, the table name is removed from the list of tables for the database.

Note: *Removing the table does not cause the table file to be removed, but only the removes the table name from the table list. You must remove the table file if you no longer need it.*

To deregister a table, select Table Registration from the Tables Menu (see 11.3) by entering **5** at the prompt. The **tqzip** program displays the following Table Registration Menu:

```
>Enter your selection <1> :
>5
>
>      Table Registration Menu
>
>    1)  Register a Table...
>    2)  Deregister a Table...
>
>   99)  Exit Table Registration Menu
```

Enter **2** at the prompt to deregister a table. The **tqzip** program displays the following prompts:

```
>Enter your selection <1> :
>2
>
>Registered tables:                                                    A
>
>1)  2200 Archive           Runs
>
>Enter Table Number :                                                  B
>1
>
>Table 2200 Archive:Runs deregistered.
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A — The **tqzip** program displays a list of the tables registered in the database. This example shows one registered table, 2200 Archive:Runs.

B — Enter the number of the table you want to deregister at the prompt.

The **tqzip** program then provides a message indicating the table that has been deregistered.

The 2200 Archive:Runs table is deregistered. The table name is removed from the list of tables in the database.

11.10. Unloading Table Specifications

The **tqzip** program allows you to export data collection and retention specifications for the tables currently contained in the performance database into a file. You can use this file to view the data collection and retention specifications and update them by using a text editor. You can then reload the table specifications by using the -l and -c options on the **tqzip** command line (see 11.11).

To unload table specifications with **tqzip**, use the following command line format:

```
@tqzip [-d,databasename,] -u,-c,tablespec-file
```

where:

-d specifies the database that **tqzip** uses. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting of the tq-ini element is used (see 3.1).

-u specifies that an unload operation is to be performed by **tqzip**. The type of unload operation is "table collection/retention" as indicated by the -c option.

-c unloads the data collection and retention specifications from the database to the file specified by the *tablespec-file* parameter.

Example

The following shows an example of unloading table specifications to the file tablefile.

```
>@tqzip -u,-c,tablefile.
>TQZIP 7R4G (011312 0704:45) Fri Jan  13 08:11:27 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>Using database TQDATA:PRODUCTION
>
>...unloading table specifications to:tablefile.
>
>Table information unloaded.
>
>End TQZIP
```

Table Specification File Format

The table specifications appear in the following format in the output file created by **tqzip**. If you choose to edit this file or update the definitions, you should maintain this file format. See 11.5 and 11.6 for details on adding or modifying collection and retention specifications.

```
BEGIN TABLESPECS unloaded on Fri Jan  13 14:33:57 2012                A
[2200:Runs]                                                           B
RetentionStart=0000
PrimaryRetention=1DAY
RetentionSpec=001000,5DAY
RetentionSpec=010000,1MONTH
Collection=ON
CollectionSpec=0000,2400,0030,0000000
FieldSpec=Orig_Runid,ON
FieldSpec=Account,ON
FieldSpec=Program,ON
FieldSpec=Pgm_Start,ON
FieldSpec=KSize,ON
FieldSpec=Run_Elap,ON
FieldSpec=Int_Elap,ON
FieldSpec=Tot_Sups,ON
FieldSpec=Int_Sups,ON
FieldSpec=Int_%Sups,ON
FieldSpec=Tot_IP,ON
FieldSpec=Int_IP,ON
FieldSpec=Tot_IO,ON
FieldSpec=Int_IO,ON
FieldSpec=Tot_ER/CC,ON
FieldSpec=Int_ER/CC,ON
FieldSpec=Tot_IOreq,ON
FieldSpec=Int_IOreq,ON
FieldSpec=Tot_IOrate,ON
FieldSpec=Int_IOrate,ON
FieldSpec=Sample_Time,ON
FieldSpec=Type,ON
FieldSpec=Count,ON
#                                                                     C
END TABLESPECS                                                        D
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A The first line of the table specification file must be the header line BEGIN TABLESPECS. It is followed by a series of table specifications.

B This line begins the first table specification.

Each table specification begins with the class:table name in brackets ([ ]). The number of lines in each set of table specifications is variable. The lines must appear in order and the label must appear exactly as shown followed by an equals (=) sign. Each of the values on a line must be separated by a comma or a space. The lines are as follows:

[class:table]	The class and table name of a table that is registered in the performance database.
RetentionStart	The synchronization time for aggregated records. All aggregated records will start on even intervals of the retention start time. RetentionStart is specified as *hhmm* and must be a value greater than or equal to 0000 and less than 2400. If no value is specified for this field, 0000 is used as the default. A value of 0000 is always used if the table does not contain intervalized data.
PrimaryRetention	The length of time the records created by the data collectors are retained. This field is specified as *nUNITS*. *n* must be an integer greater than zero and *UNITS* can be MINUTE, HOUR, DAY, WEEK, MONTH, or YEAR. A value is required for this field and it must be greater than 0.
RetentionSpec	The interval and the length of time the records created by the data collectors are retained. Zero or more lines of RetentionSpec are allowed. Each retention specification is on a separate line and is made up of two values separated by a space or a comma: RetentionSpec=*interval,retention-period* The interval is specified as *hhmmss*. The retention-period is specified as *nUNITS*. *n* must be an integer greater than zero and *UNITS* can be MINUTE, HOUR, DAY, WEEK, MONTH, or YEAR. A value is required for both *interval* and *retention-period* in each RetentionSpec.

	The set of RetentionSpec lines in the load file replaces any RetentionSpec lines previously defined for the table. If no RetentionSpec lines are specified in the load file, all existing RetentionSpec lines are deleted for the table and no new ones are added. RetentionSpec lines are ignored if the table does not contain intervalized data.
Collection	Indicates whether or not data collection is turned on for the table. If data collection is to be done for the table, specify ON. Otherwise, specify OFF. If no value is specified, ON is used as the default.
CollectionSpec	The start and stop time, the interval, and the exclusion mask for data collection. Zero or more lines of CollectionSpec are allowed. Each collection specification is on a separate line and is made up of four values separated by a space or a comma: CollectionSpec=*start,stop,interval,exclusion-mask* The *start* and *stop* values indicate the start and stop times of data collection and are specified as *hhmm*. The *interval* is specified as *mmss*. The *exclusion-mask* is specified as a series of seven digits corresponding to the days of the week, beginning with Sunday and ending with Saturday. The mask is specified using the digit 1 to exclude a day and the digit 0 to include a day. For example, the mask 1000001 indicates that Sunday and Saturday are excluded from data collection. If exclusion-mask is not specified, 1000001 is used by default. A value is required for *start, stop*, and *interval* in each CollectionSpec. The set of CollectionSpec lines in the load file replaces any CollectionSpec lines previously defined for the table. If no CollectionSpec lines are specified in the load file, all existing CollectionSpec lines are deleted for the table and no new ones are added. The *interval* specification on a CollectionSpec is ignored if the table does not contain intervalized data.

FieldSpec — Indicates whether or not a field in the table is to be collected by the TeamQuest Baseline probes. Zero or more lines of FieldSpec are allowed. Each FieldSpec is on a separate line and is made up of two values separated by a space or a comma:

FieldSpec=*field,value*

The *field* must be the name of a field that exists in the table.

The *value* can be either ON or OFF. ON indicates that the TeamQuest Baseline probes will collect data for this field. OFF indicates that no data collection will be done for this field. If no value is specified for a field, the FieldSpec line is ignored.

Fields that are part of the group key defined for the table cannot be unloaded or loaded.

If no FieldSpec line exists in the load file for a given table field, the status of the data collection flag for that field remains unchanged in the table.

`C` Comment lines begin with a # character.

`D` The last line of the file must be the END TABLESPECS line.

11.11. Loading Table Specifications

You can reload the performance database with a new set of table data collection and retention specifications by using the -l and -c options on the **tqzip** command line. You should use the file as created by the **tqzip** -u and -c options as input for loading table specifications (see 11.10). If you update the table specification file, you should ensure that the format of the file remains consistent with what was output by the **tqzip** -u and -c options.

The tables named in the file being loaded must already exist in the database. If a table does not exist, a warning error is printed and the specifications for that table are ignored. Existing table specifications for the tables being loaded are replaced with the data collection and retention information contained in the load file.

To load table specifications with **tqzip**, use the following command line format:

```
@tqzip [-d,databasename,] -l,-c,tablespec-file
```

where:

-d specifies the database that **tqzip** uses. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting of the tq-ini element is used (see 3.1).

-l specifies that a load operation is to be performed by **tqzip**. The type of load operation is "table collection/retention" as indicated by the -c option.

-c loads the data collection and retention specifications into the performance database from the file specified by the *tablespec-file* parameter. The *tablespec-file* parameter is the name of a data and retention specification file created by the -u and -c options of **tqzip** (see 11.10).

Example

The following shows an example of loading table specifications from the file tablefile to the performance database.

```
>@tqzip -l,-c,tablefile
>TQZIP 7R4G (011312 0704:45) Fri  Jan  13 08:15:27 2012
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>Using database TQDATA:PRODUCTION
>
>Table information loaded.
>
>End TQZIP
```

11.12. Reorganizing Table Files

Table data (such as process data and alarm logging) is maintained in a set of files in the performance database. The **tqrorg** utility is provided to assist in the maintenance and recovery of table files. It may become necessary to reorganize your table files when any of the following occur:

- A table has a problem as indicated by an error termination from a probe or utility with one of the following error messages:

  ```
  *Error Invalid record sentinel detected
  *Error Invalid record heading in file <filename>
  *Error Invalid record type in file <filename>
  *Error Internal error in index selection on table
  ```

- Any problems with the table index, such as a selection problem
- A probe or utility appears to be performing poorly or the file size appears to be unusually large

Every table file should be reorganized periodically, if possible. Performing the reorganization at least once a month is a good starting point. The amount of data you collect and how frequently the **tqagg** program is run can affect the frequency of reorganizing the table data.

11.12.1. Why Do I Need To Reorganize My Table Files?

A number of factors enter into the need for periodic reorganization. Because the actual reorganization can be a time-consuming process, it cannot be performed automatically on an on-going basis. A table file needs to be reorganized for the following reasons:

- As records are deleted from tables, the space is left available for reuse as new records are inserted. Fragmentation of the table space can occur over a period of time. Reorganization consolidates this fragmented space.
- When the **tqagg** program is run, it removes records from a table based on the retention information for that table. If data is collected for a long period of time without **tqagg** being run, the table file will grow in size. If the **tqagg** program is then run, the records are removed and the space used by the removed records is marked as available for reuse. This space is very likely to be in the front or middle of the file. For this reason, the actual size of the file is not reduced. If **tqagg** is run on a regular basis, from that point on the free space in the file is reused, but the file size remains constant. Executing the **tqrorg** utility causes the records to be rewritten at the front of the file and any remaining file space to be freed. This will decrease the total size of the file.
- As records are added and deleted from a table, the table manager attempts to balance the indexes for the table. However, an optimum balance cannot always be maintained. Running **tqrorg** causes the indexes to be rebuilt, ensuring that they are properly balanced. Properly balanced indexes increase the efficiency of record searching.
- In the event of an abnormal system termination, it is possible that one or more table files will be left in an inconsistent state. Running **tqrorg** can repair these problems. It is possible some of the most recently added records can be lost during this recovery process.

11.12.2. Executing tqrorg

To execute the **tqrorg** utility, use the following command line format:

```
@tqrorg table-file1. [,table-file2.,table-file3., ...]
```

where *table-file1*, *table-file2*, and *table-file3* are names of table files that exist in the performance database.

If you need to obtain the file name for a given table, use the Tables Menu from the **tqzip** program and select List Tables (see 11.4).

If you attempt to reorganize a file that is not a table file, an error message will be printed, but no damage will occur to the specified file.

If you attempt to reorganize a file that is currently open by another run, the reorganization will not be done. In the event of a system crash, the file open count can become incorrect. Before the reorganization can take place, the table manager subsystem needs to be deactivated by entering the control statement

```
@sys$lib$*solar.deact sys$lib$*tq-bslgate.tq-tmssd
```

Appendix A
OS 2200 Parameters

A.1. Introduction

This appendix contains a list of the parameters collected by the TeamQuest Baseline and TeamQuest Online probes. The parameters listed in this appendix are classified by the hierarchy of key names. The hierarchy can consist of the system, category group, category, subcategory, and statistic name. This appendix lists the parameters collected by the SIP, TPM, RTPM, MIPS, IOTRACE, MAPPER log, system log, and ODTP probes.

- SIP Parameters
- TPM Log Parameters
- MIPS Parameters
- IOTRACE Parameters
- MAPPER Log Parameters
- System Log Parameters
- Workload Performance Derived Statistics
- System Activity Monitor Statistics
- Open Distributed Transaction Processing Statistics

A.2. SIP Parameters

This subsection contains a list of the parameters collected by the SIP probe. These parameters are collected by TeamQuest Baseline and TeamQuest Online.

- System Statistics
- Processor Statistics
- Dedicated Java Processor Statistics
- Memory Statistics
- I/O Activity Statistics
- Queue Statistics

The SIP probe monitors the data in the category groups included in the following overview of the Category Group, Category, and Subcategory definitions.

Category Group	Category	Subcategory
System	Population	
	Response Time	
	Throughput	
	Exec Requests	System Total
		Total
	Exec CALLs	System Total
		Total

Category Group	Category	Subcategory
Processor	Total	Utilization
		Exec 3 Utilization
		Interrupts
		Exec Requests
		Switching
		Hardware Summary
		Collection Base
		Util by Switching Level
		Hardware Interrupts
	Processor	Collection Base
		Hardware Interrupts
		Hardware Monitor
		Interrupts
		Util by Switching Level
		Utilization

Category Group	Category	Subcategory
Processor Java	Total	Utilization
		Collection Base
		Hardware Interrupts
	Processor	Utilization
		Collection Base
		Hardware Interrupts

Category Group	Category	Subcategory
Memory	Size	
	EXPOOL	
	Program Load	Total
		TIP Mem Trans Abs
		TIP Mem Trans ZOOMs
		Non TIP Mem Trans Abs
		Non TIP Mem Trans ZOOMs
		TIP Trans Abs
		TIP Trans ZOOMs
		Demand Abs
		Demand ZOOMs
		Batch Abs
		Batch ZOOMs
	Paging Activity	Real Space Usage
		Working Set Mgmt
		Working Sets

Category Group	Category	Subcategory
IO Activity	Physical Channel	Summary
	Logical Channel	Summary
		Cache Summary
		Read Hit
		Write Hit
		Read Miss
		Write Miss
		Read Bypass
		Write Bypass
		Store Through
	Control Unit	Summary
	Disk	Summary

	Tape	Summary
	Other	Summary
	Cache Disk	Read Hit
		Write Hit
		Read Miss
		Write Miss
		Read Bypass
		Write Bypass
		Store Through
	Record Lock Processor	
	Memory File	Summary
	XPC	Read Hit
		Write Hit
		Read Miss
		Write Miss
		Destage
		Bypass
		Misc Exec
		Summary
		Utilization
	XIIP	Summary
	Mulithost	TIP
		UDS
		Summary
		UDS RLP Locks
		UDS RLP Unlocks
		UDS RLP Misc
Category Group	**Category**	**Subcategory**
Queue	System Return	Entries
		Server
		MaxCount
	Available	Entries
		MaxCount

A.2.1. System Statistics

The following statistics are the general statistics classified in the System category group.

Parameter Hierarchy

Category Group:	System
Category:	Population
Statistic Name:	
Total Runs Open	Total number of open runs
Demand Runs Open	Average number of demand runs open
Batch Runs Open	Average number of batch runs open
Demand Runs In Core	Average number of demand runs in core
Batch Runs In Core	Average number of batch runs in core
Backlogged Runs	Average number of backlogged runs
TIP Schd Queue Size	Average number of TIP transactions scheduled

Category Group:	System
Category:	Response Time
Statistic Name:	
Demand Resp Time	Average demand response time in seconds
Demand Resp < 0.5 Sec	Percent of demand responses less than 0.5 seconds
Demand Resp < 1 Sec	Percent of demand responses less than 1 second
Demand Resp < 3 Sec	Percent of demand responses less than 3 seconds
Demand Resp < 5 Sec	Percent of demand responses less than 5 seconds
Demand Resp < 10 Sec	Percent of demand responses less than 10 seconds
Batch Resp Time	Average batch response time in seconds

Category Group:	System
Statistic Name:	
Elapsed Time	Time in seconds for the sample
Active IPs	Current number of “non Java” Instruction Processors that are active
Active Java IPs	Current number of Java Instruction Processors that are active
ES SUP Secs	Exec service SUP charges by non-TIP programs (in seconds)
ES SUP TIP Secs	Exec service SUP charges by TIP transactions and batch connect programs (in seconds)

CS SUP Secs	Control statement SUP charges by non-TIP programs (in seconds)
CS SUP TIP Secs	Control statement SUP charges by TIP transactions and batch connect programs (in seconds)
IP SUP Secs	IP SUP charges by non-TIP instruction processor execution (in seconds
IP SUP TIP Secs	IP SUP charges by TIP instruction processor execution for TIP transactions and batch connect programs (in seconds)
USER IO SUP Secs	User I/O SUP charges to non-TIP files (in seconds)
USER IO SUP TIP Secs	User I/O SUP charges to TIP files (in seconds

Category Group:	System
Category:	Throughput
Statistic Name:	
TIP Txns/Sec	Number of TIP transactions scheduled per second
TIP Txns/Sec MIN	Minimum number of TIP transactions scheduled per second
TIP Txns/Sec MAX	Maximum number of TIP transactions scheduled per second
TIP Txns Completed/Sec	Number of TIP transactions completed per second
Demand Resps/Sec	Number of demand responses per second
Batch Runs Opened /Sec	Number of batch runs opened per second
Demand Runs Opened	Total demand runs opened
Batch Runs Opened	Total batch runs opened
Total Runs Opened	Total of newly opened runs

Category Group:	System
Category:	Exec Requests
Subcategory:	System Total
Statistic Name:	
Reqs/Sec	Requests per second for all Executive Requests (ER)

Category Group:	System
Category:	Exec CALLs
Subcategory:	System Total
Statistic Name:	
Reqs/Sec	Requests per second for all call gates defined on the system

Category Group:	System
Category:	Exec Requests
Resource:	IO$, DATE$, ...
Subcategory:	Total
Statistic Name:	
Reqs/Sec	Requests per second for each Executive Request (ER)

Category Group:	System
Category:	Exec CALLs
Resource:	CREATE$GATE, DELETE$GATE, ...
Subcategory:	Total
Statistic Name:	
Reqs/Sec	Requests per second for each call gate

A.2.2. Processor Statistics

The Processor category group provides summary statistics for general processor, Hardware Monitor, and Performance Monitor utilization.

The following statistics are available on all systems with SIP level 4 or higher configured.

Parameter Hierarchy

Category Group:	Processor
Category:	Total
Subcategory:	Switching
Statistic Name:	
Avg Queue Size	Average number of activities waiting to be dispatched
Queue Searches	Number of times the switching queue was searched

The following statistics are available only on 2200 systems with SIP level 5 configured.

Parameter Hierarchy

Category Group:	Processor
Category:	Total
Subcategory:	Util by Switching level
Statistic Name:	
Exec 0	Time the processor was executing in Exec 0
Exec 1	Time the processor was executing in Exec 1
Exec 3	Time the processor was executing in Exec 3

SIP	Time the processor was executing in SIP
TIP	Time the processor was executing in TIP
Real Time	Time the processor was executing in real time
Deadline	Time the processor was executing in deadline batch
Demand	Time the processor was executing in demand
Batch	Time the processor was executing in batch

Category Group:	Processor
Category:	Processor
Resource:	IP0, IP1, ...
Subcategory:	Util by Switching Level
Statistic Name:	
Exec 0	Time the processor was executing in Exec 0
Exec 1	Time the processor was executing in Exec 1
Exec 3	Time the processor was executing in Exec 3
SIP	Time the processor was executing in SIP
TIP	Time the processor was executing in TIP
Real Time	Time the processor was executing in real time
Deadline	Time the processor was executing in deadline batch
Demand	Time the processor was executing in demand
Batch	Time the processor was executing in batch

A.2.2.1. Performance Monitor

The following statistics are available on 2200 systems only.

Parameter Hierarchy

Category Group:	Processor
Category:	Total
Subcategory:	Collection Base
Statistic Name:	
Available Time	Total available time for all IP performance monitors for this sample
Available %	Percent all IP performance monitors were available for this sample

Category Group:	Processor
Category:	Total
Subcategory:	Utilization
Statistic Name:	
Idle	Percent of time idle including forced idle
Forced Idle	Percent of tme in forced idle
Idle MIN	Minimum percent of time idle
Idle MAX	Maximum percent of time idle
Total Exec	Percent of time in Exec processing
Total Exec MIN	Minimum percent of time in Exec processing
Total Exec MAX	Maximum percent of time in Exec processing
Total User	Percent of time in user processing
Total User MIN	Minimum percent of time in user processing
Total User MAX	Maximum percent of time in user processing
Misc NonSwitchable	Percentage of miscellaneous nonswitchable Exec processing
IO Processing	Percentage of Exec I/O processing
Processor Control	Percentage of Exec processor control processing
Memory Management	Percentage of Exec memory management
TIP/Integ Rec/Aud Cntrl	Percentage of Exec TIP, integrated recovery, and audit control processing
Symbionts	Percentage of Exec symbionts processing
Misc Switchable	Percentage of miscellaneous switchable Exec processing
Real Time	Percentage of user real-time processing
ILE	Percentage of independently linked Exec subsystems
CMS 1100	Percentage of CMS 1100 processing
UDS	Percentage of UDS processing

TIP	Percentage of TIP processing
TIP MIN	Minimum percentage of TIP processing
TIP MAX	Maximum percentage of TIP processing
Critical Deadline	Percentage of critical deadline processing
Demand	Percentage of demand processing
Batch	Percentage of batch processing
SCP Factor	Software Controlled Performance level factor for the system
SCP Scaled Factor	Scaled System controlled Performance level factor for the system
Performance Level %	Contracted performance level percent for the system

Category Group:	Processor
Category:	Total
Subcategory:	Hardware Interrupts
Statistic Name:	
Hardware Default	Hardware default interrupts per second
Unretryable Hwr Chk	Unretryable hardware check interrupts per second
Diagnostic	Diagnostic interrupts per second
Interrupt 3	Interrupt 3 (not used) per second
Interrupt 4	Interrupt 4 (not used) per second
Interrupt 5	Interrupt 5 (not used) per second
Interrupt 6	Interrupt 6 (not used) per second
Interrupt 7	Interrupt 7 (not used) per second
Reference Violation	Reference violation interrupts per second
Addressing Exception	Addressing interrupts
Terminal Addr Exception	Terminal addressing exception interrupts per second
RSC/Underflow /Overflow	RCS/generic stack underflow/overflow interrupts per second
Signal	Signal interrupts per second
Test and Set	Test-and-set interrupts per second
Invalid Instruction	Invalid instruction interrupts per second
Page Exception	Page exception interrupts per second
Arithmetic Exception	Arithmetic exception interrupts per second
Data Exception	Data exception interrupts per second
Operation Trap	Operation trap interrupts per second
Breakpoint	Breakpoint interrupts per second
Quantum Timer	Quantum timer interrupts per second
Interrupt 21	Interrupt 21 (not used) interrupts per second

Interrupt 22	Interrupt 22 (not used) interrupts per second
Interrupt 23	Interrupt 23 (not used) interrupts per second
Software Break	Software break interrupts per second
Jump History Full	Jump history full interrupts per second
Delayed Hardware Check	Delayed hardware check interrupts per second
Dayclock	Dayclock interrupts per second
Performance Monitoring	Performance monitoring interrupts per second
Initial Program Load	Initial program load interrupts per second
UPI Initial	UPI initial interrupts per second
UPI Normal	UPI normal interrupts per second

Category Group: Processor
Category: Processor
Resource: IP0, IP1, ...
Subcategory: Collection Base
Statistic Name:

Available Time	Available time for this IP performance monitor for this sample
Available %	Percent that this IP performance monitor was available for this sample

Category Group: Processor
Category: Processor
Resource: IP0, IP1, ...
Subcategory: Utilization
Statistic Name:

Idle	Percentage of time idle
Total Exec	Percentage of time in Exec processing
Total User	Percentage of time in user processing
Misc NonSwitchable	Percentage of miscellaneous nonswitchable
IO Processing	Percentage of Exec I/O processing
Processor Control	Percentage of Exec processor control processing
Memory Management	Percentage of Exec memory management processing
TIP/Integ Rec/Aud Cntrl	Percentage of Exec TIP, integrated recovery, and audit control processing
Symbiont	Percentage of Exec symbionts processing
Misc Switchable	Percentage of miscellaneous switchable Exec processing
Real Time	Percentage of user real-time processing

ILES	Percentage of independently linked Exec subsystems
CMS 1100	Percentage of CMS 1100 processing
UDS	Percentage of UDS processing
TIP	Percentage of TIP processing
Critical Deadline	Percentage of critical deadline processing
Demand	Percentage of demand processing
Batch	Percentage of batch processing

Category Group:	Processor
Category:	Processor
Resource:	IP0, IP1, ...
Subcategory:	Hardware Interrupts
Statistic Name:	
Hardware Default	Hardware default interrupts per second
Unretryable Hwr Chk	Unretryable hardware check interrupts per second
Diagnostic	Diagnostic interrupts per second
Interrupt 3	Interrupt 3 (not used) interrupts
Interrupt 4	Interrupt 4 (not used) interrupts
Interrupt 5	Interrupt 5 (not used) interrupts
Interrupt 6	Interrupt 6 (not used) interrupts
Interrupt 7	Interrupt 7 (not used) interrupts
Reference Violation	Reference violation interrupts per second
Addressing Exception	Addressing interrupts
Terminal Addr Exception	Terminal addressing exception interrupts per second
RSC/Underflow /Overflow	RCS/generic stack underflow/overflow interrupts per second
Signal	Signal interrupts per second
Test and Set	Test and set interrupts per second
Invalid Instruction	Invalid instruction interrupts per second
Page Exception	Page exception interrupts per second
Arithmetic Exception	Arithmetic exception interrupts per second
Data Exception	Data exception interrupts per second
Operation Trap	Operation trap interrupts per second
Breakpoint	Breakpoint interrupts per second
Quantum Timer	Quantum timer interrupts per second
Interrupt 21	Interrupt 21 (not used) interrupts per second
Interrupt 22	Interrupt 22 (not used) interrupts per second
Interrupt 23	Interrupt 23 (not used) interrupts per second

Software Break	Software break interrupts per second
Jump History Full	Jump history full interrupts per second
Delayed Hardware Check	Delayed hardware check interrupts per second
Dayclock	Dayclock interrupts per second
Performance Monitoring	Performance monitoring interrupts per second
Initial Program Load	Initial program load interrupts per second
UPI Initial	UPI initial interrupts per second
UPI Normal	UPI normal

A.2.2.2. Dedicated Java Processor Statistics

The following statistics are available on systems where it is possible to designate a processor or set of processors as dedicated Java IPs.

Parameter Hierarchy

Category Group:	Processor Java
Category:	Total
Subcategory:	Collection Base
Statistic Name:	
Available Time	Total available time for all IP performance monitors for this sample
Available %	Percent all IP performance monitors were available for this sample

Category Group:	Processor Java
Category:	Total
Subcategory:	Utilization
Statistic Name:	
Idle	Percent of time idle
Idle MIN	Minimum percent of time idle
Idle MAX	Maximum percent of time idle
Total Exec	Percent of time in Exec processing
Total Exec MIN	Minimum percent of time in Exec processing
Total Exec MAX	Maximum percent of time in Exec processing
Total User	Percent of time in user processing
Total User MIN	Minimum percent of time in user processing
Total User MAX	Maximum percent of time in user processing
Misc NonSwitchable	Percentage of miscellaneous nonswitchable Exec processing
IO Processing	Percentage of Exec I/O processing
Processor Control	Percentage of Exec processor control processing
Memory Management	Percentage of Exec memory management
TIP/Integ Rec /Aud Cntrl	Percentage of Exec TIP, integrated recovery, and audit control processing
Symbionts	Percentage of Exec symbionts processing
Misc Switchable	Percentage of miscellaneous switchable Exec processing
Real Time	Percentage of user real-time processing
ILES	Percentage of independently linked Exec subsystems
CMS 1100	Percentage of CMS 1100 processing
UDS	Percentage of UDS processing

TIP	Percentage of TIP processing
TIP MIN	Minimum percentage of TIP processing
TIP MAX	Maximum percentage of TIP processing
Critical Deadline	Percentage of critical deadline processing
Demand	Percentage of demand processing
Batch	Percentage of batch processing

Category Group:	Processor Java
Category:	Total
Subcategory:	Hardware Interrupts
Statistic Name:	
Hardware Default	Hardware default interrupts per second
Unretryable Hwr Chk	Unretryable hardware check interrupts per second
Diagnostic	Diagnostic interrupts per second
Interrupt 3	Interrupt 3 (not used) per second
Interrupt 4	Interrupt 4 (not used) per second
Interrupt 5	Interrupt 5 (not used) per second
Interrupt 6	Interrupt 6 (not used) per second
Interrupt 7	Interrupt 7 (not used) per second
Reference Violation	Reference violation interrupts per second
Addressing Exception	Addressing interrupts
Terminal Addr Exception	Terminal addressing exception interrupts per second
RSC/Underflow /Overflow	RCS/generic stack underflow/overflow interrupts per second
Signal	Signal interrupts per second
Test and Set	Test-and-set interrupts per second
Invalid Instruction	Invalid instruction interrupts per second
Page Exception	Page exception interrupts per second
Arithmetic Exception	Arithmetic exception interrupts per second
Data Exception	Data exception interrupts per second
Operation Trap	Operation trap interrupts per second
Breakpoint	Breakpoint interrupts per second
Quantum Timer	Quantum timer interrupts per second
Interrupt 21	Interrupt 21 (not used) interrupts per second
Interrupt 22	Interrupt 22 (not used) interrupts per second
Interrupt 23	Interrupt 23 (not used) interrupts per second
Software Break	Software break interrupts per second
Jump History Full	Jump history full interrupts per second

Delayed Hardware Check	Delayed hardware check interrupts per second
Dayclock	Dayclock interrupts per second
Performance Monitoring	Performance monitoring interrupts per second
Initial Program Load	Initial program load interrupts per second
UPI Initial	UPI initial interrupts per second
UPI Normal	UPI normal interrupts per second

Category Group: Processor Java
Category: Processor
Resource: IP0, IP1, ...
Subcategory: Collection Base
Statistic Name:

Available Time	Available time for this IP performance monitor for this sample
Available %	Percent that this IP performance monitor was available for this sample

Category Group: Processor Java
Category: Processor
Resource: IP0, IP1, ...
Subcategory: Utilization
Statistic Name:

Idle	Percentage of time idle
Total Exec	Percentage of time in Exec processing
Total User	Percentage of time in user processing
Misc NonSwitchable	Percentage of miscellaneous nonswitchable
IO Processing	Percentage of Exec I/O processing
Processor Control	Percentage of Exec processor control processing
Memory Management	Percentage of Exec memory management processing
TIP/Integ Rec /Aud Cntrl	Percentage of Exec TIP, integrated recovery, and audit control processing
Symbiont	Percentage of Exec symbionts processing
Misc Switchable	Percentage of miscellaneous switchable Exec processing
Real Time	Percentage of user real-time processing
ILES	Percentage of independently linked Exec subsystems
CMS 1100	Percentage of CMS 1100 processing
UDS	Percentage of UDS processing
TIP	Percentage of TIP processing

Critical Deadline	Percentage of critical deadline processing
Demand	Percentage of demand processing
Batch	Percentage of batch processing

Category Group:	Processor Java
Category:	Processor
Resource:	IP0, IP1, ...
Subcategory:	Hardware Interrupts
Statistic Name:	
Hardware Default	Hardware default interrupts per second
Unretryable Hwr Chk	Unretryable hardware check interrupts per second
Diagnostic	Diagnostic interrupts per second
Interrupt 3	Interrupt 3 (not used) interrupts
Interrupt 4	Interrupt 4 (not used) interrupts
Interrupt 5	Interrupt 5 (not used) interrupts
Interrupt 6	Interrupt 6 (not used) interrupts
Interrupt 7	Interrupt 7 (not used) interrupts
Reference Violation	Reference violation interrupts per second
Addressing Exception	Addressing interrupts
Terminal Addr Exception	Terminal addressing exception interrupts per second
RSC/Underflow /Overflow	RCS/generic stack underflow/overflow interrupts per second
Signal	Signal interrupts per second
Test and Set	Test and set interrupts per second
Invalid Instruction	Invalid instruction interrupts per second
Page Exception	Page exception interrupts per second
Arithmetic Exception	Arithmetic exception interrupts per second
Data Exception	Data exception interrupts per second
Operation Trap	Operation trap interrupts per second
Breakpoint	Breakpoint interrupts per second
Quantum Timer	Quantum timer interrupts per second
Interrupt 21	Interrupt 21 (not used) interrupts per second
Interrupt 22	Interrupt 22 (not used) interrupts per second
Interrupt 23	Interrupt 23 (not used) interrupts per second
Software Break	Software break interrupts per second
Jump History Full	Jump history full interrupts per second
Delayed Hardware Check	Delayed hardware check interrupts per second
Dayclock	Dayclock interrupts per second

Performance Monitoring	Performance monitoring interrupts per second
Initial Program Load	Initial program load interrupts per second
UPI Initial	UPI initial interrupts per second
UPI Normal	UPI normal

A.2.3. Memory Statistics

The Memory Category Group provides general summary, program load, and paging statistics.

The memory utilization statistics are not available from the Exec on OS 2200 on EM64T Server systems. As the statistics will exist in the SIP data file (with a value of 0), TeamQuest Baseline will report zeros in the display of this data.

A.2.3.1. Program Load Statistics

The following statistics are available for Exec level SB4R3 and higher.

Parameter Hierarchy

Category Group:	Memory
Category:	Program Load
Subcategory:	Total
Statistic Name:	
Init Program Size	Average initial program size in K words
Init Program Loads /Sec	Total initial program loads per second
Init Bank Size	Average initial bank size in K words
Init Bank Loads/Sec	Total initial bank loads per second
Dyn Bank Size	Average dynamic bank size in K words
Dyn Bank Loads/Sec	Total dynamic bank loads per second

Category Group:	Memory
Category:	Program Load
Subcategory:	Demand Abs
Statistic Name:	
Init Program Size	Average initial program size in K words
Init Program Loads /Sec	Total initial program loads per second
Init Bank Size	Average initial bank size in K words

Init Bank Loads/Sec	Total initial bank loads per second
Dyn Bank Size	Average dynamic bank size in K words
Dyn Bank Loads/Sec	Total dynamic bank loads per second

Category Group:	Memory
Category:	Program Load
Subcategory:	Demand ZOOMs
Statistic Name:	
Init Program Size	Average initial program size in K words
Init Program Loads /Sec	Total initial program loads per second
Init Bank Size	Average initial bank size in K words
Init Bank Loads/Sec	Total initial bank loads per second
Dyn Bank Size	Average dynamic bank size in K words
Dyn Bank Loads/Sec	Total dynamic bank loads per second

Category Group:	Memory
Category:	Program Load
Subcategory:	Batch Abs
Statistic Name:	
Init Program Size	Average initial program size in K words
Init Program Loads /Sec	Total initial program loads per second
Init Bank Size	Average initial bank size in K words
Init Bank Loads/Sec	Total initial bank loads per second
Dyn Bank Size	Average dynamic bank size in K words
Dyn Bank Loads/Sec	Total dynamic bank loads per second

Category Group:	Memory
Category:	Program Load
Subcategory:	Batch ZOOMs
Statistic Name:	
Init Program Size	Average initial program size in K words
Init Program Loads /Sec	Total initial program loads per second
Init Bank Size	Average initial bank size in K words
Init Bank Loads/Sec	Total initial bank loads per second
Dyn Bank Size	Average dynamic bank size in K words
Dyn Bank Loads/Sec	Total dynamic bank loads per second

The following statistics are available from Exec level SB4R3 through level SB5R4.

Parameter Hierarchy

Category Group:	Memory
Category:	Program Load
Subcategory:	TIP Mem Trans Abs
Statistic Name:	
Init Program Size	Average initial program size in K words
Init Program Loads /Sec	Total initial program loads per second
Init Bank Size	Average initial bank size in K words
Init Bank Loads/Sec	Total initial bank loads per second
Dyn Bank Size	Average dynamic bank size in K words
Dyn Bank Loads/Sec	Total dynamic bank loads per second

Category Group:	Memory
Category:	Program Load
Subcategory:	TIP Mem Trans ZOOMs
Statistic Name:	
Init Program Size	Average initial program size in K words
Init Program Loads /Sec	Total initial program loads per second
Init Bank Size	Average initial bank size in K words
Init Bank Loads/Sec	Total initial bank loads per second
Dyn Bank Size	Average dynamic bank size in K words
Dyn Bank Loads/Sec	Total dynamic bank loads per second

Category Group:	Memory
Category:	Program Load
Subcategory:	Non TIP Mem Trans Abs
Statistic Name:	
Init Program Size	Average initial program size in K words
Init Program Loads /Sec	Total initial program loads per second
Init Bank Size	Average initial bank size in K words
Init Bank Loads/Sec	Total initial bank loads per second
Dyn Bank Size	Average dynamic bank size in K words
Dyn Bank Loads/Sec	Total dynamic bank loads per second

Category Group:	Memory
Category:	Program Load
Subcategory:	Non TIP Mem Trans ZOOMs
Statistic Name:	
Init Program Size	Average initial program size in K words
Init Program Loads /Sec	Total initial program loads per second
Init Bank Size	Average initial bank size in K words
Init Bank Loads/Sec	Total initial bank loads per second
Dyn Bank Size	Average dynamic bank size in K words
Dyn Bank Loads/Sec	Total dynamic bank loads per second

The following statistics are available for Exec level SB6 and higher.

Parameter Hierarchy

Category Group:	Memory
Category:	Program Load
Subcategory:	TIP Trans Abs
Statistic Name:	
Init Program Size	Average initial program size in K words
Init Program Loads /Sec	Total initial program loads per second
Init Bank Size	Average initial bank size in K words
Init Bank Loads/Sec	Total initial bank loads per second
Dyn Bank Size	Average dynamic bank size in K words
Dyn Bank Loads/Sec	Total dynamic bank loads per second

Category Group:	Memory
Category:	Program Load
Subcategory:	TIP Trans ZOOMS
Statistic Name:	
Init Program Size	Average initial program size in K words
Init Program Loads /Sec	Total initial program loads per second
Init Bank Size	Average initial bank size in K words
Init Bank Loads/Sec	Total initial bank loads per second
Dyn Bank Size	Average dynamic bank size in K words
Dyn Bank Loads/Sec	Total dynamic bank loads per second

A.2.4. Paging Statistics

The memory utilization statistics are not available from the Exec on OS 2200 on EM64T Server systems. As the statistics will exist in the SIP data file (with a value of 0), TeamQuest Baseline will report zeros in the display of this data.

The following statistics are available for the 2200 systems only.

Parameter Hierarchy

Category Group:	Memory
Category:	Paging Activity
Statistic Name:	
Locked Pages	Average number of locked pages
Single Page Loads/Sec	Single page loads caused by memory tight conditions, pages per second
Queued Page Loads /Sec	Number of page load requests queued because real memory is unavailable, requests per second
Queued Pages/Sec	Number of pages queued because real memory unavailable, pages per second
ACW Group Tracks /Sec	Total number of tracks written for working set output, tracks per second
ACW Groups/Sec	Count of access control word (ACW) groups, ACWs per second

Category Group:	Memory
Category:	Paging Activity
Subcategory:	Real Space Usage
Statistic Name:	
Nailed Pages	Pages of real memory that are nailed, in pages
Active Pages	Pages of real memory that are active, in pages
Free Pages	Pages of real memory that are free, in pages
Free Pages MIN	Minimum pages of real memory that are free, in pages
Free Pages MAX	Maximum pages of real memory that are free, in pages
Reserved Pages	Pages of real memory that are reserved, in pages
Down Pages	Pages of real memory that are down, in pages
Verify/Recovering Pages	Pages of real memory that are verifying or recovering, in pages
Exec Pages	Pages of real memory used by Exec, in pages
User Pages	Pages of real memory used by users, in pages
File Pages	Pages of real memory used by files, in pages
TIP	Pages of real memory used by users of TIP in pages
Shared	Pages of real memory used by users, shared in pages

Demand	Pages of real memory used by users of demand in pages
Batch	Pages of real memory used by users of batch, in pages

Category Group:	Memory
Category:	Paging Activity
Subcategory:	Working Set Mgmt
Statistic Name:	
Imm Page Faults/Sec	Page faults that did not deactivate the faulting process, pages per second
Delayed Page Faults /Sec	Delayed page faults where the faulting process was suspended, pages per second
Delayed with IO/Sec	Delayed page faults that had I/O, pages per second
Avg IO Time per Delay	I/O time per delayed page fault, milliseconds per page fault
Avg Size User Queue	Average number of pages on the page load request queue

Category Group:	Memory
Category:	Paging Activity
Subcategory:	Working Sets
Statistic Name:	
Inputs/Sec	Working sets input, working sets per second
Pages Input/Sec	Pages of working set input, pages per second
Outputs/Sec	Working sets output, working sets per second
Pages Output/Sec	Pages of working sets output, pages per second
IO Time per Output	Average I/O time per working set output, millisecond per working set

A.2.5. I/O Activity Statistics

The I/O Activity category group provides general statistics for I/O activity.

Parameter Hierarchy

Category Group:	IO Activity
Statistic Name:	
Total IOs/Sec	Total I/Os per second
Total IOs/Sec MIN	Minimum total I/Os per second
Total IOs/Sec MAX	Maximum total I/Os per second
Disk IOs/Sec	Total disk I/Os per second
Tape IOs/Sec	Total tape I/Os per second
Avg RET-FLD	Average service time
Avg KWords/Req	Number of K words per request
Avg Xfer Time	Average transfer time
KWords/Sec	Average K words per second

Category Group:	IO Activity
Category:	Physical Channel
Resource:	PC0, PC1, ...
Subcategory:	Summary
Statistic Name:	
Percent Busy	Physical channel percent busy
Reqs/Sec	Physical channel request per second
KWords/Req	Physical channel K words per request
KWords/Sec	Physical channel K words per second
Transfer Time	Physical channel transfer time in milliseconds

Category Group:	IO Activity
Category:	Logical Channel
Resource:	LC0, LC1
Subcategory:	Summary
Statistic Name:	
Percent Busy	Logical channel percent busy
Reqs/Sec	Logical channel request per second
Percent Reqs Read	Logical channel percent requests that were reads
Percent Reqs Queued	Logical channel percent of requests that were queued
KWords/Req	Logical channel K words per request
KWords/Req In	Logical channel K words per input request
KWords/Req Out	Logical channel K words per output request

Hit Rate	Logical channel hit rate
Transfer Time	Logical channel transfer time
RET-FLD	Logical channel service time
Hardware Serv Time	Logical channel hardware service time
Concurrency	Logical channel concurrency

Category Group:	IO Activity
Category:	Logical Channel
Resource:	LC0, LC1
Subcategory:	Cache Summary
Statistic Name:	
Reqs/Sec	Logical channel request per second
KWords/Req	Logical channel K words per request
Hit Rate	Logical channel hit rate
Transfer Time	Logical channel transfer time
RET-FLD	Logical channel service time
Hardware Serv Time	Logical channel hardware service time
Speculation Success	Logical channel speculation success
Current NSRI	Current value of number of segments rolled in (NSRI)

Category Group:	IO Activity
Category:	Logical Channel
Resource:	LC0, LC1
Subcategory:	Read Hit
Statistic Name:	
Reqs/Sec	Logical channel read hit requests per second
KWords/Req	Logical channel read hit K words per request
Transfer Time	Logical channel read hit transfer time
RET-FLD	Logical channel read hit service time
Hardware Serv Time	Logical channel read hit hardware service time
% of All Reqs	Logical channel read hit percent of all requests

Category Group:	IO Activity
Category:	Logical Channel
Resource:	LC0, LC1
Subcategory:	Write Hit
Statistic Name:	
Reqs/Sec	Logical channel write hit requests per second
KWords/Req	Logical channel write hit K words per request

Transfer Time	Logical channel write hit transfer time
RET-FLD	Logical channel write hit service time
Hardware Serv Time	Logical channel write hit hardware service time
% of All Reqs	Logical channel write hit percent of all requests

Category Group:	IO Activity
Category:	Logical Channel
Resource:	LC0, LC1
Subcategory:	Read Miss
Statistic Name:	
Reqs/Sec	Logical channel read miss requests per second
KWords/Req	Logical channel read miss K words per request
Transfer Time	Logical channel read miss transfer time
RET-FLD	Logical channel read miss service time
Hardware Serv Time	Logical channel read miss hardware service time
% of All Reqs	Logical channel read miss percent of all requests

Category Group:	IO Activity
Category:	Logical Channel
Resource:	LC0, LC1
Subcategory:	Write Miss
Statistic Name:	
Reqs/Sec	Logical channel write miss requests per second
KWords/Req	Logical channel write miss K words per request
Transfer Time	Logical channel write miss transfer time
RET-FLD	Logical channel write miss service time
Hardware Serv Time	Logical channel write miss hardware service time
% of All Reqs	Logical channel write miss percent of all requests

Category Group:	IO Activity
Category:	Logical Channel
Resource:	LC0, LC1
Subcategory:	Read Bypass
Statistic Name:	
Reqs/Sec	Logical channel read bypass requests per second
KWords/Req	Logical channel read bypass K words per request
Transfer Time	Logical channel read bypass transfer time

RET-FLD	Logical channel read bypass service time
Hardware Serv Time	Logical channel read bypass hardware service time
% of All Reqs	Logical channel read bypass percent of all requests

Category Group:	IO Activity
Category:	Logical Channel
Resource:	LC0, LC1
Subcategory:	Write Bypass
Statistic Name:	
Reqs/Sec	Logical channel write bypass requests per second
KWords/Req	Logical channel write bypass K words per request
Transfer Time	Logical channel write bypass transfer time
RET-FLD	Logical channel write bypass service time
Hardware Serv Time	Logical channel write bypass hardware service time
% of All Reqs	Logical channel write bypass percent of all requests

Category Group:	IO Activity
Category:	Logical Channel
Resource:	LC0, LC1
Subcategory:	Store Through
Statistic Name:	
Reqs/Sec	Logical channel store through requests per second
KWords/Req	Logical channel store through K words per request
Transfer Time	Logical channel store through transfer time
RET-FLD	Logical channel store through service time
Hardware Serv Time	Logical channel store through hardware service time
% of All Reqs	Logical channel store through percent of all requests

Category Group:	IO Activity
Category:	Control Unit
Resource:	CU0, CU1
Subcategory:	Summary
Statistic Name:	
Percent Busy	Control unit percent busy
Reqs/Sec	Control unit request per second
Percent Reqs Read	Percentage of all requests that were reads
Percent Reqs Queued	Percentage of all requests that were queued
KWords/Req	Number of K words per request
KWords/Req In	Number of K words per request that were input

KWords/Req Out	Number of K words per request that were output
Hit Rate	Hit rate
Transfer Time	Control unit transfer time
RET-FLD	Control unit service time
Hardware Serv Time	Control unit hardware service time
Concurrency	Control unit concurrency
Response Time	Unit average response time in milliseconds
KWords/Sec	K words transferred per second
Avg Queue Length	Average number of requests queued to this unit

Category Group:	IO Activity
Category:	Disk
Resource:	DSK0, DSK1
Subcategory:	Summary
Statistic Name:	
Percent Busy	Disk percent busy
Reqs/Sec	Number of requests per second
Percent Reqs Read	Percentage of all requests that were reads
Percent Reqs Queued	Percentage of all requests that were queued
KWords/Req	Number of K words per request
KWords/Req In	Number of K words per request that were input
KWords/Req Out	Number of K words per request that were output
Hit Rate	Hit rate
Transfer Time	Average transfer time for a request
RET-FLD	Average service time
Hardware Serv Time	Average hardware service time
Concurrency	Concurrency
Response Time	Unit average response time in milliseconds
KWords/Sec	K words transferred per second
Avg Queue Length	Average number of requests queued to this unit
Channel Queue Time	Channel queuing time in milliseconds

Category Group:	IO Activity
Category:	Tape
Resource:	TAP0, TAP1
Subcategory:	Summary
Statistic Name:	
Percent Busy	Tape percent busy
Reqs/Sec	Number of requests per second

Percent Reqs Read	Percentage of all requests that were reads
Percent Reqs Queued	Percentage of all requests that were queued
KWords/Req	Average number of K words per request
KWords/Req In	Average number of K words per request that were input
KWords/Req Out	Average number of K words per request that were output
Transfer Time	Average transfer time per request
RET-FLD	Average service time
Hardware Serv Time	Average hardware service time
Concurrency	Concurrency
Response Time	Unit average response time in milliseconds
KWords/Sec	K words transferred per second
Avg Queue Length	Average number of requests queued to this unit
Channel Queue Time	Channel queuing time in milliseconds

Category Group:	IO Activity
Category:	Other
Resource:	DCP0, PRNT1
Subcategory:	Summary
Statistic Name:	
Percent Busy	Device percent busy
Reqs/Sec	Number of requests per second
Percent Reqs Read	Percentage of all requests that were reads
Percent Reqs Queued	Percentage of all requests that were queued
KWords/Req	Average number of K words per request
KWords/Req In	Average number of K words per request that were input
KWords/Req Out	Average number of K words per request that were output
Transfer Time	Average transfer time per request
RET-FLD	Average service time
Hardware Serv Time	Average hardware service time
Concurrency	Concurrency

Category Group:	IO Activity
Category:	Cache Disk
Statistic Name:	
Destage Reqs/Sec	Cache disk destage requests per second
Destage Avoided	Cache disk destage avoided
Speculation Success	Cache disk speculation success
Disk Traffic	Cache disk traffic
Hit Rate	Cache disk hit rate

Age Oldest Segment	Cache disk age of the oldest segment
Reqs/Sec	Cache disk request per second
KWords/Req	Cache disk K words per request
Transfer Time	Cache disk transfer time
RET-FLD	Cache disk service time
Hardware Serv Time	Cache disk hardware service time

Category Group:	IO Activity
Category:	Cache Disk
Subcategory:	Read Hit
Statistic Name:	
Reqs/Sec	Cache disk read hit requests per second
KWords/Req	Cache disk read hit K words per request
Transfer Time	Cache disk read hit transfer time
RET-FLD	Cache disk read hit service time
Hardware Serv Time	Cache disk read hit hardware service time
Percent of All Reqs	Cache disk read hit percent of all requests

Category Group:	IO Activity
Category:	Cache Disk
Subcategory:	Write Hit
Statistic Name:	
Reqs/Sec	Cache disk write hit requests per second
KWords/Req	Cache disk write hit K words per request
Transfer Time	Cache disk write hit transfer time
RET-FLD	Cache disk write hit service time
Hardware Serv Time	Cache disk write hit hardware service time
Percent of All Reqs	Cache disk write hit percent of all requests

Category Group:	IO Activity
Category:	Cache Disk
Subcategory:	Read Miss
Statistic Name:	
Reqs/Sec	Cache disk read miss requests per second
KWords/Req	Cache disk read miss K words per request
Transfer Time	Cache disk read miss transfer time
RET-FLD	Cache disk read miss service time
Hardware Serv Time	Cache disk read miss hardware service time
Percent of All Reqs	Cache disk read miss percent of all requests

Category Group:	IO Activity
Category:	Cache Disk
Subcategory:	Write Miss
Statistic Name:	
Reqs/Sec	Cache disk write miss requests per second
KWords/Req	Cache disk write miss K words per request
Transfer Time	Cache disk write miss transfer time
RET-FLD	Cache disk write miss service time
Hardware Serv Time	Cache disk write miss hardware service time
Percent of All Reqs	Cache disk write miss percent of all requests

Category Group:	IO Activity
Category:	Cache Disk
Subcategory:	Read Bypass
Statistic Name:	
Reqs/Sec	Cache disk read bypass requests per second
KWords/Req	Cache disk read bypass K words per request
Transfer Time	Cache disk read bypass transfer time
RET-FLD	Cache disk read bypass service time
Hardware Serv Time	Cache disk read bypass hardware service time
Percent of All Reqs	Cache disk read bypass percent of all requests

Category Group:	IO Activity
Category:	Cache Disk
Subcategory:	Write Bypass
Statistic Name:	
Reqs/Sec	Cache disk write bypass requests per second
KWords/Req	Cache disk write bypass K words per request
Transfer Time	Cache disk write bypass transfer time
RET-FLD	Cache disk write bypass service time
Hardware Serv Time	Cache disk write bypass hardware service time
Percent of All Reqs	Cache disk write bypass percent of all requests

Category Group:	IO Activity
Category:	Cache Disk
Subcategory:	Store Through
Statistic Name:	
Reqs/Sec	Cache disk store through requests per second
KWords/Req	Cache disk store through K words per request

Transfer Time	Cache disk store through transfer time
RET-FLD	Cache disk store through service time
Hardware Serv Time	Cache disk store through hardware service time
Percent of All Reqs	Cache disk store through percent of all requests

Category Group:	IO Activity
Category:	Record Lock Processor
Statistic Name:	
Lock Reqs/Sec	Total lock requests per second
Unlock Reqs/Sec	Total unlock requests per second
Delayed Resp Msg/Sec	Total delayed response messages per second
Deadlocks/Sec	Total deadlocks per second
Lock Reqs	Total lock requests
Unlock Reqs	Total unlock requests
Delayed Resp Msg	Total delayed response messages
Deadlocks	Total deadlocks

Category Group:	IO Activity
Category:	Memory File
Subcategory:	Summary
Statistic Name:	
Reqs/Sec	Memory file requests per second
Transfer Time	Transfer time in milliseconds for memory files
Delay Time	File lock delay time in milliseconds for memory files
RET	Request existence time in milliseconds for memory files
Percent Busy	Percentage of time that memory file data was transferred
KWords/Sec	Number of K words per second transferred for memory files
KWords/Req	Number of K words per I/O request for memory files
KWords/Req In	Number of K words per read I/O request for memory files
KWords/Req Out	Number of K words per write I/O request for memory files
Percent Reqs Read	Percentage of all memory file accesses that were read requests

The following statistics are created if the Exec is level SB4R8 and you have an Extended Processing Complex (XPC) with the Exec ETCEXEC feature installed.

Parameter Hierarchy

Category Group:	IO Activity
Category:	Multihost
Subcategory:	Summary
Statistic Name:	
Lock Reqs/Sec	Total lock requests per second
Unlock Reqs/Sec	Total unlock requests per second
Delayed Resp Msg/Sec	Total delayed response messages per second
Deadlocks/Sec	Total deadlocks per second
Lock Reqs	Total lock requests
Unlock Reqs	Total unlock requests
Delayed Resp Msg	Total delayed response messages
Deadlock	Total deadlocks

Category Group:	IO Activity
Category:	Multihost
Subcategory:	TIP
Statistic Name:	
Lock Reqs/Sec	Total lock requests per second by TIP
Unlock Reqs/Sec	Total unlock requests per second by TIP
Delayed Resp Msg/Sec	Total delayed response messages per second by TIP
Deadlocks/Sec	Total deadlocks per second by TIP
Lock Reqs	Total lock requests by TIP
Unlock Reqs	Total unlock requests by TIP
Delayed Resp Msg	Total delayed response messages by TIP
Deadlocks	Total deadlocks encountered by TIP

Category Group:	IO Activity
Category:	Multihost
Subcategory:	UDS
Statistic Name:	
Lock Reqs/Sec	Total lock requests per second by UDS
Unlock Reqs/Sec	Total unlock requests per second by UDS
Delayed Resp Msg/Sec	Total delayed response messages per second by UDS
Deadlocks/Sec	Total deadlocks per second by UDS

Lock Reqs	Total lock requests by UDS
Unlock Reqs	Total unlock requests by UDS
Delayed Resp Msg	Total delayed response messages by UDS
Deadlock	Total deadlocks encountered by UDS

The following statistics are created for UDS when you have an XPC and the UDSTRP SIP counter.

Parameter Hierarchy

Category Group:	IO Activity
Category:	Multihost
Subcategory:	UDS RLP Locks
Statistic Name:	
Lock Rec	Total record locks by UDS
Lock Blk	Total block locks by UDS
Lock File	Total file locks by UDS
Maint Lock	Total maintenance locks by UDS
Immed Lock Rec	Total immediate record locks by UDS
Immed Lock Blk	Total immediate block locks by UDS
Lock String	Total string locks by UDS
Lock Blk Rec Prb	Total block, record, and probe validity locks by UDS
Lock Rec Prb	Total record and probe validity locks by UDS
Lock Rec/Sec	Total record locks per second by UDS
Lock Blk/Sec	Total block locks per second by UDS
Lock File/Sec	Total file locks per second by UDS
Maint Lock/Sec	Total maintenance locks per second by UDS
Immed Lock Rec/Sec	Total immediate record locks per second by UDS
Immed Lock Blk/Sec	Total immediate block locks per second by UDS
Lock String/Sec	Total string locks per second by UDS
Lock Blk Rec Prb/Sec	Total block, record, and probe validity locks per second by UDS
Lock Rec Prb/Sec	Total record and probe validity locks per second by UDS

Category Group:	IO Activity
Category:	Multihost
Subcategory:	UDS RLP Unlocks
Statistic Name:	
Unlock Rec	Total record unlocks by UDS
Unlock Blk	Total block unlocks by UDS
Unlock Proc	Total process unlocks by UDS

Unlock Subcoll	Total subcollection unlocks by UDS
Unlock File	Total file unlocks by UDS
Unlock String	Total string unlocks by UDS
Release Blk Rec	Total release block record unlocks by UDS
Unlock Rec/Sec	Total record unlocks per second by UDS
Unlock Blk/Sec	Total block unlocks per second by UDS
Unlock Proc/Sec	Total process unlocks per second by UDS
Unlock Subcoll/Sec	Total subcollection unlocks per second by UDS
Unlock File/Sec	Total file unlocks per second by UDS
Unlock String/Sec	Total string unlocks per second by UDS
Release Blk Rec/Sec	Total release block record unlocks per second by UDS

Category Group: IO Activity

Category: Multihost

Subcategory: UDS RLP Misc

Statistic Name:

RLP Req	Total number of UDS request to the RLP
Switch Port	Total switch ports by UDS
Init Dump	Total initial dumps by UDS
Purge Node	Total purge nodes by UDS
Down Subcoll	Total down subcollections by UDS
Prb Valid	Total probe validity checks by UDS
Write Valid	Total write validity checks by UDS
Dec Subcoll Cache	Total declare subcollection caches by UDS
Chg Owner Coll	Total change owner collection changes by UDS
RLP Req/Sec	Total number of UDS requests to the RLP per second by UDS
Switch Port/Sec	Total switch ports per second by UDS
Init Dump/Sec	Total initial dumps per second by UDS
Purge Node/Sec	Total purge nodes per second by UDS
Down Subcoll/Sec	Total down subcollections per second by UDS
Prb Valid/Sec	Total probe validity checks per second by UDS
Write Valid/Sec	Total write validity checks per second by UDS
Dec Subcoll Cache/Sec	Total declare subcollection caches per second by UDS
Chg Owner Coll/Sec	Total change owner collection changes per second by UDS

A.2.6. SIP Parameters for the XPC

The following is a list of the statistics collected by the SIP probe for the XPC.

Parameter Hierarchy

Category Group:	IO Activity
Category:	XPC
Resource:	XPC0, XPC1
Subcategory:	Summary
Statistic Name:	
Reqs/Sec	XPC requests per second
Words/Req	XPC words per request
Avg RET	XPC average request existence time (ms)
Function Rsp Time	XPC average response time for all XPC functions (ms)
Exec Staged Words/Sec	XPC number of words per second staged by the Exec
Exec Destage Words/Sec	XPC number of words per second destaged by the Exec
Discarded Dst Reqs/S	XPC initiated destage requests per second that were discarded
Destage Avoided Reqs/S	XPC destage requests per second that were avoided
Disk Traffic %	Percent of disk traffic for XPC requests in comparison to a disk-only environment
Hit Rate	XPC hit rate as a percentage
XIIP Queue Length	Average initiation queue length of all XIIPs connected to this XPC
Programs/Sec	Average XPC programs per second executed by the XPC
Program Rsp Time	Average XPC response time in milliseconds for an XPC program
XPC-L UD Synch Reqs/Sec	Average number of XPC-L UD Synch entry write requests per second

Category Group:	IO Activity
Category:	XPC
Resource:	XPC0, XPC1
Subcategory:	Utilization
Statistic Name:	
IXP	Average percentage of time that each IXP is busy
Available	Percentage of time the XPC Performance Monitor is available
Written to File Storage	Average percentage of XPC file acceleration storage that has been written to file storage

Category Group:	IO Activity
Category:	XPC
Resource:	XPC0, XPC1
Subcategory:	Read Hit
Statistic Name:	
Reqs/Sec	XPC read hit requests per second
Words/Req	XPC read hit words per request
Avg RET	XPC read hit average request existence time
Response Time	XPC read hit average response time

Category Group:	IO Activity
Category:	XPC
Resource:	XPC0, XPC1
Subcategory:	Write Hit
Statistic Name:	
Reqs/Sec	XPC write hit requests per second
Words/Req	XPC write hit words per request
Avg RET	XPC write hit average request existence time
Response Time	XPC write hit average response time

Category Group:	IO Activity
Category:	XPC
Resource:	XPC0, XPC1
Subcategory:	Read Miss
Statistic Name:	
Reqs/Sec	XPC read miss requests per second
Words/Req	XPC read miss words per request
Avg RET	XPC read miss average request existence time
Response Time	XPC read miss average response time

Category Group:	IO Activity
Category:	XPC
Resource:	XPC0, XPC1
Subcategory:	Write Miss
Statistic Name:	
Reqs/Sec	XPC write miss requests per second
Words/Req	XPC write miss words per request
Avg RET	XPC write miss average request existence time
Response Time	XPC write miss average response time

Category Group:	IO Activity
Category:	XPC
Resource:	XPC0, XPC1
Subcategory:	Destage
Statistic Name:	
Reqs/Sec	XPC destage requests per second
Words/Req	XPC destage words per request
Avg RET	XPC destage average request existence time
Response Time	XPC destage average response time

Category Group:	IO Activity
Category:	XPC
Resource:	XPC0, XPC1
Subcategory:	Bypass
Statistic Name:	
Reqs/Sec	XPC bypass requests per second
Words/Req	XPC bypass words per request
Avg RET	XPC bypass average request existence time
Response Time	XPC bypass average response time

Category Group:	IO Activity
Category:	XPC
Resource:	XPC0, XPC1
Subcategory:	Misc Exec
Statistic Name:	
Reqs/Sec	XPC miscellaneous Exec requests per second
Words/Req	XPC miscellaneous Exec per request
Avg RET	XPC miscellaneous Exec request existence time
Response Time	XPC miscellaneous Exec response time

Category Group:	IO Activity
Category:	XIIP
Resource:	XIIP0, XIIP1
Subcategory:	Summary
Statistic Name:	
Programs/Sec	XPC programs per second on this XIIP
Avg Queue Length	XIIP initiation queue length
Program Rsp Time	XPC response time for programs from this XIIP
Lock Reqs/Sec	XPC-L lock function requests per second from this XIIP
Lock Process Time	Average XPC-L lock command processing time in milliseconds from this XIIP

A.2.7. Queue Statistics

The following statistics are collected by the SIP probe for the 2200 Queueing Architecture.

Parameter Hierarchy

Category Group:	Queue
Category:	System Return
Subcategory:	Entries
Statistic Name:	
Enqueued/Sec	Number of queue entries enqueued per second to the system return queue
Queued Time (msec)	Time in milliseconds per entry spent on the system return queue

Category Group:	Queue
Category:	System Return
Subcategory:	Server
Statistics Name:	
Activations/Sec	Rate at which the system return queue server is activated

Category Group:	Queue
Category:	System Return
Subcategory:	MaxCount
Statistic Name:	
Exceeded/Sec	Number of MaxCount exceeded interrupts per second for the system return queue

Category Group:	Queue
Category:	Available
Subcategory:	Entries
Statistic Name:	
Enqueued/Sec	Rate at which entries are enqueued to the available queue
Queued Time (msec)	Time in milliseconds per entry spent on any available queue

Category Group:	Queue
Category:	Available
Subcategory:	MaxCount
Statistic Name:	
Expansions/Sec	Number of MaxCount expansions per second for any available queue
Contractions/Sec	Number of MaxCount contractions per second for any available queue

Category Group:	Queue
Category:	Available
Statistic Name:	
Deactivations/Sec	Rate at which activities are deactivated on any available queue
Queues	Number of available queues in the system

Category Group:	Queue
Statistic Name:	
Banks	Number of banks in the system

A.3. TPM Log Parameters

This subsection contains a list of the parameters collected by the TeamQuest Baseline TPM and RTPM log probes. The following statistics are collected for each workload defined on the system.

Note: *These parameters are not collected by TeamQuest Online.*

Parameter Hierarchy

Category Group:	HVSTAT Log
Category:	Workload
Subcategory:	Total
Statistic Name:	
Total SUPs	Total standard units of processing (SUP)
IP SUPs	Total IP SUPs
ER/CC SUPs	Total ER and control card SUPs
I/O SUPs	Total I/O SUPs
Lock Wait Time	Total time spent waiting on locks
Lock Held Time	Total time spent holding file control superstructure (FCSS) locks
Processing Time	Total amount of time spent processing
Response Time	Total amount of time spent to complete a transaction
Termination Time	The average commit duration time
Initial Loads	The number of transaction initial loads and block transfer loads
Initial Load Time	The total amount of time between program initial load and initiation for TIP initial loads and block transfers.
Stuck Copies	Total number of stuck transactions with same transaction code
Total Stuck	Total number of stuck transactions in the whole system
RTPS Deact Copies	Total number of RTPS deact copies with same transaction code
Total RTPS Deact	Total number of RTPS deactivated transactions in the whole system
Active Copies	Number of transactions active with same transaction code
Total Active	Total number of transactions active in the whole system
Txns Scheduled	Total number of transactions scheduled in the whole system
Txns Completed	Total number of transactions completed in the whole system
Passoff Msgs	Total number of user to user passoff messages
Output Messages	Total number of output messages sent

FCSS Read Func Reqs	Total number of FCSS read function requests
FCSS Write Func Reqs	Total number of FCSS write function requests
FCSS Lock Func Reqs	Total number of FCSS lock function requests
FCSS Other Func Reqs	Total number of FCSS other function requests
FCSS Read Words	Total number of words read by FCSS read requests
FCSS Write Words	Total number of words written by FCSS write requests
FCSS Locks Acquired	Total number of locks acquired
Exec Reads	Total number of Exec reads
Exec Writes	Total number of Exec writes
TIP Reads	Total number of TIP reads
TIP Writes	Total number of TIP writes
Transaction Count	Total number of transactions
I/O Requests	Total number of I/O requests for this transaction
Txns/sec System	Average number of transactions per second for the whole system

A.4. MIPS Parameters

This subsection contains a list of the parameters collected by the TeamQuest Baseline **tqmips** probe for Capacity On Demand (COD) systems.

Note: *These parameters are not collected by TeamQuest Online.*

Parameter Hierarchy

Category Group:	MIPS
Category:	Total
Subcategory:	Utilization
Statistic Name:	
Avg MIPS Used	Average number of MIPS used
MIPS Seconds	Total number of MIPS seconds used
MIPS Interval	Total number of second for the reporting period
Desired MIPS	The number of desired MIPS for the partition
Baseline MIPS	The number of baseline MIPS for the system
Ceiling MIPS	The number of ceiling MIPS for the system
Running MIPS	Total number of running MIPS for the partition

A.4.1. MIPS Probe Derived Statistics

The following statistics are collected by the MIPS (**tqmips**) probe.

Parameter Hierarchy

Category Group:	MIPS
Category:	Total
Subcategory:	Utilization
Statistic Name:	
Desired Percent	Percentage of the Desired MIPS used
Baseline Percent	Percentage of the Baseline MIPS used
Ceiling Percent	Percentage of the Ceiling MIPS used
Running Percent	Percentage of the Running MIPS used
MIPS Months	Number of MIPS-Months used

A.5. IOTRACE Parameters

This subsection contains a list of the parameters collected by the TeamQuest Baseline IOTRACE probe. The statistics are collected for each workload defined on the system.

Note: *These parameters are not collected by TeamQuest Online.*

The following statistics are collected by workload set, workload, category group, category, subcategory, and statistic.

Parameter Hierarchy

Category Group:	IOTRACE
Category:	Workload
Subcategory:	Total
Statistic Name:	
Requests	Number of I/O requests
Read Requests	I/O requests that were reads
Words/Req In	Words per input request
Words/Req Out	Words per output request
Cached Requests	Number of I/O request that were cached
RET	Time required by all system devices to process the I/O (file lock delay + chan/deviceQ + hardware serv time)
Hardware Serv Time	Time required by the device to process the I/O (seek + latency + CU delay + transfer time)
Transfer Time	Time required to transfer the data

Category Group:	IOTRACE
Category:	Workload
Resource:	LC0, LC1
Subcategory:	Logical Channel
Statistic Name:	
Requests	Number of I/O requests
Read Requests	I/O requests that were reads
Words/Req In	Words per input request
Words/Req Out	Words per output request
Cached Requests	Number of I/O request that were cached
RET	Time required by all system devices to process the I/O (file lock delay + chan/deviceQ + hardware serv time)
Hardware Serv Time	Time required by the device to process the I/O (seek + latency + CU delay + transfer time)
Transfer Time	Time required to transfer the data

A.6. MAPPER Log Parameters

This subsection contains a list of the parameters collected by the TeamQuest Baseline MAPPER log probe. This probe collects data based on the defined workload sets.

Note: *These parameters are not collected by TeamQuest Online.*

The following statistics are collected by workload set, workload, category group, category, subcategory, and statistic.

Parameter Hierarchy

Category Group:	Mapper Log
Category:	Workload
Subcategory:	Total
Statistic Name:	
I/Os Processed	Number of input and output operations processed
Lines Processed	Number of lines read and written
Breakpoint	Number of breakpoints that occurred
Run Functions	Number of run functions that occurred
Display Functions	Number of display functions that occurred
Connect Time	Total time in seconds that stations were connected
Active Time (ms)	Total function active time in milliseconds
Avg Sec Response	Number of seconds per logged function
Functions Logged	Number of logged functions that occurred
Non Run Funcs Logged	Number of logged functions that occurred outside of runs
Functions Started	Number of MAPPER functions that were processed
Funcs Under Run Logged	Number of logged functions that occurred under runs
Funcs Under Run Started	Number of started functions that occurred under runs
Run Lines Processed	Number of lines read and written from functions under runs
Run I/Os	Number of input and output operations from functions under runs
Run Read I/Os	Number of input operations from functions under runs
Interactions	Number of user terminal interactions that occurred
Response (Sec/Int)	Number of seconds per interaction
Run Interactions	Number of user terminal interactions during runs that occurred

Non-Run Interactions	Number of manual user interactions that occurred
Response (Sec/Run Int)	Number of seconds per run interaction
Response (S/Non-RunInt)	Number of seconds per manual interaction

A.7. System Log Parameters

This subsection contains a list of the parameters collected by the TeamQuest Baseline system log probe. The following statistics are collected for each workload defined on the system.

Note: *These parameters are not collected by TeamQuest Online.*

Parameter Hierarchy

Category Group:	System Log
Category:	Workload
Subcategory:	Total
Statistic Name:	
Total SUPs	Total SUPs used to process the job. Sum of the IP SUPs, ER/CC SUPs, operating system SUPs, and I/O SUPs.
IP SUPs	Total IP SUPs
ER/CC SUPs	Total ER and control card SUPs
ES IO SUPs	Total Exec Services I/O SUPs
ES IP SUPs	Total Exec Services IP SUPs
CS IO SUPs	Total control statement I/O SUPs
CS IP SUPs	Total control statement IP SUPs
IO SUPs	Total I/O SUPs
Core Block SUPs	Total core block SUPs. Sum of SUPs and main storage usage.
IO Requests	Total number of I/O requests made by this run
Disk I/O Requests	Total number of I/O requests to disk that have been made by this run
Disk I/O Words	Total number of words that have been transferred during disk I/O operations by this run
Tape I/O Requests	Total number of I/O requests to tape that have been made by this run
Tape I/O Words	Total number of words that have been transferred during tape I/O operations by this run
Voluntary Wait	Total voluntary wait time. Sum of user response time and TWAIT$ time.
Resource Wait	Total amount of time the run spent waiting for a resource
Active Time	Total time the job was active

Transactions	Total number of transactions
Cards In	Total number of cards the Exec read during the run
Runs Opened	Total number of runs that were opened
Max Real Memory	Maximum number of real pages concurrently resident for this run
Real Memory Usage	The product of the total number of resident pages and the time in seconds that the pages are resident

Category Group: System Log
Category: Workload
Subcategory: TIP
Statistic Name:

Total SUPs	Total SUPs used to process the job. Sum of the IP SUPs, ER/CC SUPs, operating system SUPs, and I/O SUPs.
IP SUPs	Total IP SUPs
ER/CC SUPs	Total ER and control card SUPs
ES IO SUPs	Total Exec Services I/O SUPs
ES IP SUPs	Total Exec Services IP SUPs
CS IO SUPs	Total control statement I/O SUPs
CS IP SUPs	Total control statement IP SUPs
IO SUPs	Total I/O SUPs
Core Block SUPs	Total core block SUPs. Sum of SUPs and main storage usage.
IO Requests	Total number of I/O requests made by this transaction
Disk I/O Requests	Total number of I/O requests to disk that have been made by this transaction
Disk I/O Words	Total number of words that have been transferred during disk I/O operations by this transaction
Tape I/O Requests	Total number of I/O requests to tape that have been made by this transaction
Tape I/O Words	Total number of words that have been transferred during tape I/O operations by this transaction
Voluntary Wait	Total voluntary wait time. Sum of user response time and TWAIT$ time.
Resource Wait	Total amount of time the transaction spent waiting for a resource
Active Time	Total time the transaction was active
Transactions	Total number of transactions
Max Real Memory	Maximum number of real pages concurrently resident for this transaction
Real Memory Usage	The product of the total number of resident pages and the time in seconds that the pages are resident

Category Group:	System Log
Category:	Workload
Subcategory:	Demand
Statistic Name:	
Total SUPs	Total SUPs used to process the job. Sum of the IP SUPs, ER/CC SUPs, operating system SUPs, and I/O SUPs.
IP SUPs	Total IP SUPs
ER/CC SUPs	Total ER and control card SUPs
IO SUPs	Total I/O SUPs
ES IO SUPs	Total Exec Services I/O SUPs
ES IP SUPs	Total Exec Services IP SUPs
CS IO SUPs	Total control statement I/O SUPs
CS IP SUPs	Total control statement IP SUPs
Core Block SUPs	Total core block SUPs. Sum of SUPs and main storage usage.
IO Requests	Total number of I/O requests made by this run
Disk I/O Requests	Total number of I/O requests to disk that have been made by this run
Disk I/O Words	Total number of words that have been transferred during disk I/O operations by this run
Tape I/O Requests	Total number of I/O requests to tape that have been made by this run
Tape I/O Words	Total number of words that have been transferred during tape I/O operations by this run
Voluntary Wait	Total voluntary wait time. Sum of user response time and TWAIT$ time.
Resource Wait	Total amount of time the run spent waiting for a resource
Active Time	Total time the job was active
Cards In	Total number of cards the Exec read during the run
Runs Opened	Total number of runs that were opened
Max Real Memory	Maximum number of real pages concurrently resident for this run
Real Memory Usage	The product of the total number of resident pages and the time in seconds that the pages are resident

Category Group:	System Log
Category:	Workload
Subcategory:	Batch
Statistic Name:	
Total SUPs	Total SUPs used to process the job. Sum of the IP SUPs, ER/CC SUPs, operating system SUPs, and I/O SUPs.
IP SUPs	Total IP SUPs
ER/CC SUPs	Total ER and control card SUPs
ES IO SUPs	Total Exec Services I/O SUPs
ES IP SUPs	Total Exec Services IP SUPs
CS IO SUPs	Total control statement I/O SUPs
CS IP SUPs	Total control statement IP SUPs
IO SUPs	Total I/O SUPs
Core Block SUPs	Total core block SUPs. Sum of SUPs and main storage usage.
IO Requests	Total number of I/O requests made by this run
Disk I/O Requests	Total number of I/O requests to disk that have been made by this run
Disk I/O Words	Total number of words that have been transferred during disk I/O operations by this run
Tape I/O Requests	Total number of I/O requests to tape that have been made by this run
Tape I/O Words	Total number of words that have been transferred during tape I/O operations by this run
Voluntary Wait	Total voluntary wait time. Sum of user response time and TWAIT$ time.
Resource Wait	Total amount of time the run spent waiting for a resource
Active Time	Total time the job was active
Cards In	Total number of cards the Exec read during the run
Runs Opened	Total number of runs that were opened
Max Real Memory	Maximum number of real pages concurrently resident for this run
Real Memory Usage	The product of the total number of resident pages and the time in seconds that the pages are resident

Category Group:	System Log
Category:	Workload
Subcategory:	Deadline
Statistic Name:	
Total SUPs	Total SUPs used to process the job. Sum of the IP SUPs, ER/CC SUPs, operating system SUPs, and I/O SUPs.
IP SUPs	Total IP SUPs
ER/CC SUPs	Total ER and control card SUPs
ES IO SUPs	Total Exec Services I/O SUPs
ES IP SUPs	Total Exec Services IP SUPs
CS IO SUPs	Total control statement I/O SUPs
CS IP SUPs	Total control statement IP SUPs
IO SUPs	Total I/O SUPs
Core Block SUPs	Total core block SUPs. Sum of SUPs and main storage usage.
IO Requests	Total number of I/O requests made by this run
Disk I/O Requests	Total number of I/O requests to disk that have been made by this run
Disk I/O Words	Total number of words that have been transferred during disk I/O operations by this run
Tape I/O Requests	Total number of I/O requests to tape that have been made by this run
Tape I/O Words	Total number of words that have been transferred during tape I/O operations by this run
Voluntary Wait	Total voluntary wait time
Resource Wait	Total amount of time the run spent waiting for a resource
Active Time	Total time the job was active
Cards In	Total number of cards the Exec read during the run
Runs Opened	Total number of runs that were opened
Max Real Memory	Maximum number of real pages concurrently resident for this run
Real Memory Usage	The product of the total number of resident pages and the time in seconds that the pages are resident

A.8. Workload Performance Derived Statistics

TeamQuest Baseline maintains workload performance derived statistics that use data from the SIP (**tqbsp**), MAPPER log (**tqmpr**), and system log (**tqsys**) probes. The workload performance reports reference these statistics. Before using the workload performance reports, you must first ensure these derived statistic definitions are in your database. You can load the required derived statistic definitions using the following command:

```
@tqzip -l,-e,-s,sys$lib$*tq-baseline.derived-wpr
```

The following derived statistics are provided:

- SIP Probe Derived Statistics
- MAPPER Log Probe Derived Statistics
- System Log Probe Derived Statistics

A.8.1. SIP Probe Derived Statistics

The following statistics are collected by the SIP (**tqbsp**) probe.

Parameter Hierarchy

Category Group:	Processor
Category:	Total
Subcategory:	Utilization
Statistic Name:	
All	Overall processor utilization

Category Group:	Processor
Category:	Total (Scaled)
Subcategory:	Utilization
Statistic Name:	
Idle	Scaled percentage of time idle
Total Exec	Scaled percentage of time in Exec processing
Total User	Scaled percentage of time in user processing
Misc NonSwitchable	Scaled percentage of miscellaneous nonswitchable
IO Processing	Scaled percentage of Exec I/O processing
Processor Control	Scaled percentage of Exec processor control processing
Memory Management	Scaled percentage of Exec memory management processing
TIP/Integ Rec /Aud Cntrl	Scaled percentage of Exec TIP, integrated recovery, and audit control processing
Symbionts	Scaled percentage of Exec symbionts processing

Misc Switchable	Scaled percentage of miscellaneous switchable Exec processing
Real Time	Scaled percentage of user real-time processing
ILES	Scaled percentage of independently linked Exec subsystems
CMS 1100	Scaled percentage of CMS 1100 processing
UDS	Scaled percentage of UDS processing
TIP	Scaled percentage of TIP processing
Critical Deadline	Scaled percentage of critical deadline processing
Demand	Scaled percentage of demand processing
Batch	Scaled percentage of batch processing

Category Group:	Memory
Category:	Paging Activity
Subcategory:	Real Space Usage
Statistic Name:	
Memory Utilization	Memory utilization for XPA systems

A.8.2. MAPPER Log Probe Derived Statistics

The following statistics are collected by the MAPPER log (**tqmpr**) probe.

Parameter Hierarchy

Category Group:	Mapper Log
Category:	Overall
Subcategory:	
Workload Set:	WLS1, WLS2, ...
Workload:	
Statistic Name:	
IOs processed/sec	I/Os processed per second
Lines processed/sec	Logic and data lines processed per second
Response (sec/display)	Active seconds per display function start
Throughput (displays/sec)	Display function starts per second
Population (signed on)	Average number of signed-on users

Category Group:	Mapper Log
Category:	Workload
Subcategory:	Total
Workload Set:	WLS1, WLS2, ...
Workload:	[all]
Statistic Name:	
IOs processed/sec	I/Os processed per second
Lines processed/sec	Logic and data lines processed per second
Response (sec/display)	Active seconds per display function start
Throughput (displays/sec)	Display function starts per second
Population (signed on)	Average number of signed-on users

A.8.3. System Log Probe Derived Statistics

The following statistics are collected by the system log (**tqsys**) probe.

Parameter Hierarchy

Category Group:	System Log
Category:	Overall
Subcategory:	TIP
Workload Set:	WLS1, WLS2, ...
Workload:	
Statistic Name:	
Kmem/txn (XPA)	K words of memory per TIP transaction for XPA systems
Tape IOs/sec	TIP tape I/Os per second
Disk IOs/sec	TIP disk I/Os per second
IO SUPs/sec	TIP I/O SUPs per second
ERCC SUPs/sec	TIP ER/CC SUPs per second
IP SUPs/sec	TIP IP SUPs per second
Response (sec/txn)	Seconds per TIP transaction
Throughput (txns/sec)	TIP transactions per second
Population (active txns)	Average number of active transactions

Category Group:	System Log
Category:	Workload
Subcategory:	TIP
Workload Set:	WLS1, WLS2, ...
Workload:	[all]
Statistic Name:	
Kmem/txn (XPA)	K words of memory per TIP transaction for XPA systems
Tape IOs/sec	TIP tape I/Os per second
Disk IOs/sec	TIP disk I/Os per second
IO SUPs/sec	TIP I/O SUPs per second
ERCC SUPs/sec	TIP ER/CC SUPs per second
IP SUPs/sec	TIP IP SUPs per second
Response (sec/txn)	Seconds per TIP transaction
Throughput (txns/sec)	TIP transactions per second
Population (active txns)	Average number of active transactions

Category Group:	System Log
Category:	Overall
Subcategory:	Demand
Workload Set:	WLS1, WLS2, ...
Workload:	
Statistic Name:	
Kmem/run (XPA)	K words of memory per demand run for XPA systems
Tape IOs/sec	Demand tape I/Os per second
Disk IOs/sec	Demand disk I/Os per second
IO SUPs/sec	Demand I/O SUPs per second
ERCC SUPs/sec	Demand ER/CC SUPs per second
IP SUPs/sec	Demand IP SUPs per second
Response (sec/req)	Seconds per demand request
Throughput (reqs/sec)	Demand requests per second
Population (open runs)	Average demand runs open

Category Group:	System Log
Category:	Workload
Subcategory:	Demand
Workload Set:	WLS1, WLS2, ...
Workload:	[all]
Statistic Name:	
Kmem/run (XPA)	K words of memory per demand run for XPA systems
Tape IOs/sec	Demand tape I/Os per second
Disk IOs/sec	Demand disk I/Os per second
IO SUPs/sec	Demand I/O SUPs per second
ERCC SUPs/sec	Demand ER/CC SUPs per second
IP SUPs/sec	Demand IP SUPs per second
Response (sec/req)	Seconds per demand request
Throughput (reqs/sec)	Demand requests per second
Population (open runs)	Average demand runs open

Category Group:	System Log
Category:	Overall
Subcategory:	Deadline
Workload Set:	WLS1, WLS2, ...
Workload:	
Statistic Name:	
Kmem/run (XPA)	K words of memory per deadline batch run for XPA systems
Tape IOs/sec	Deadline batch tape I/Os per second
Disk IOs/sec	Deadline batch disk I/Os per second
IO SUPs/sec	Deadline batch I/O SUPs per second
ERCC SUPs/sec	Deadline batch ER/CC SUPs per second
IP SUPs/sec	Deadline batch IP SUPs per second
Response (sec/run)	Seconds per deadline batch run
Throughput (runs/hour)	Deadline batch runs per hour
Population (open runs)	Average deadline batch runs open

Category Group:	System Log
Category:	Workload
Subcategory:	Deadline
Workload Set:	WLS1, WLS2, ...
Workload:	[all]
Statistic Name:	
Kmem/run (XPA)	K words of memory per deadline batch run for XPA systems
Tape IOs/sec	Deadline batch tape I/Os per second
Disk IOs/sec	Deadline batch disk I/Os per second
IO SUPs/sec	Deadline batch I/O SUPs per second
ERCC SUPs/sec	Deadline batch ER/CC SUPs per second
IP SUPs/sec	Deadline batch IP SUPs per second
Response (sec/run)	Seconds per deadline batch
Throughput (runs/hour)	Deadline batch runs per hour
Population (open runs)	Average deadline batch runs open

Category Group:	System Log
Category:	Overall
Subcategory:	Batch
Workload Set:	WLS1, WLS2, ...
Workload:	
Statistic Name:	
Kmem/run (XPA)	K words of memory per batch run for XPA systems
Tape IOs/sec	Batch tape I/Os per second
Disk IOs/sec	Batch disk I/Os per second
IO SUPs/sec	Batch I/O SUPs per second
ERCC SUPs/sec	Batch ER/CC SUPs per second
IP SUPs/sec	Batch IP SUPs per second
Response (sec/run)	Seconds per batch run
Throughput (runs/hour)	Batch runs per hour
Population (open runs)	Average batch runs open

Category Group:	System Log
Category:	Workload
Subcategory:	Batch
Workload Set:	WLS1, WLS2, ...
Workload:	[all]
Statistic Name:	
Kmem/run (XPA)	K words of memory per batch run for XPA systems
Tape IOs/sec	Batch tape I/Os per second
Disk IOs/sec	Batch disk I/Os per second
IO SUPs/sec	Batch I/O SUPs per second
ERCC SUPs/sec	Batch ER/CC SUPs per second
IP SUPs/sec	Batch IP SUPs per second
Response (sec/run)	Seconds per batch
Throughput (runs/hour)	Batch runs per hour
Population (open runs)	Average batch runs open

A.9. System Activity Monitor Statistics

This subsection contains a list of the data items collected by the TeamQuest Baseline system activity monitor probe (**tqsam**). The information collected by the probe is stored in two tables:

- Runs Table
- Transaction Table

Note: *These data items are not collected by TeamQuest Online.*

A.9.1. Runs Table

The following information is stored in the performance database tables:

Table Field Hierarchy

Class:	2200
Name:	Runs
File name:	samfile
Collection interval:	Based on the table collection interval
Default retention:	1 day at primary retention specification

Data Item	Type	Description
Run_ID	String	The unique generated run-id of the run
Orig_Runid	String	The original run-id of the run
Tot_Sups	Double	Cumulative SUPs for this run since inception
Int_Sups	Double	SUPs for this run in this interval
Int_%Sups	Double	Percentage of total SUPs used by this run in this interval
Program	String	The name of the program the run is executing
KSize	Integer	The current size in K words of the run
Account	String	The account-id of the run
Type	String	Type of run. It can be one of the following: demand (demand) demand-rt (demand real-time) batch (batch) batch-rt (batch real-time) tip (tip)
Time	Double	The time this information was recorded in the database for this sample
Sample_Time	Double	The actual time that this sample was taken for the run

Interval	Double	The desired sample interval in this collection period
System	String	The system from which this sample was taken
Run_Start	Double	The time the run started
Run_Elap	Double	The total elapsed time for the run since the run started
Int_Elap	Double	The interval elapsed time since the last sample for the run
Pgm_Start	Double	The time the program started
Tot_IP	Double	Cumulative IP SUPs for this run since inception
Int_IP	Double	IP SUPs for this run in this interval
Tot_IO	Double	Cumulative I/O SUPs for this run since inception
Int_IO	Double	IO SUPs for this run in this interval
Tot_ER/CC	Double	Cumulative ER/CC SUPs for the run since inception
Int_ER/CC	Double	ER/CC SUPs for this run in this interval
Tot_IOreq	Double	Cumulative I/O requests for this run since inception
Int_IOreq	Double	Interval I/O requests for this run in this interval
Tot_IOrate	Double	Total I/Os per second for this run since inception
Int_IOrate	Double	Interval I/Os per second for this run in this interval
Count	Integer	The number of combined occurrences

A.9.2. Transaction Table

The following information is stored in the performance database tables:

Table Field Hierarchy

Class:	2200
Name:	Transactions
File name:	samfile
Collection interval:	Based on the table collection interval
Default retention:	1 day at primary retention specification

Data Item	Type	Description
Run_ID	String	The generated run-id of the transaction
Transaction	String	The name of the transaction program being executed
Elapsed	Double	The elapsed time from when the transaction program started to the current time

Tot_Sups	Double	Total SUPs accumulated for this transaction program since inception
KSize	Integer	The current size in K words used by the transaction
Type	String	The type of transaction program being executed. It can be one of the following: self-d (self-destructive) self-i (self-initializing) reent (reentrant) onl-bat (online-batch) hv-tip (high-volume TIP)
Start	Double	The time when the transaction program started
Time	Double	The time this information was recorded in the database for this sample
Sample_Time	Double	The actual time that this sample was taken for the transaction
Interval	Double	The desired sample interval in this collection period
System	String	The system from which this sample was taken
Tot_IP	Double	Total IP SUPs accumulated for this transaction program since inception
Tot_IO	Double	Total I/O SUPs accumulated for this transaction program since inception
Tot_ER/CC	Double	Total ER/CC SUPs accumulated for this transaction program since inception
Tot_IOreq	Double	Total I/O requests accumulated for this transaction program since inception
Count	Integer	The number of combined occurrences

A.10. Open Distributed Transaction Processing Statistics

This subsection contains a list of the data items collected by the TeamQuest Baseline Open Distributed Transaction Processing probe (**tqodtp**). The information collected by this probe is stored in the following tables:

- System Table Information
- Server Table Information
- Client Table Information
- Service Table Information

Note: *These data items are not collected by TeamQuest Online.*

A.10.1. System Table Information

The following information is stored in the performance database tables:

Table Field Hierarchy

Class:	2200
Name:	ODTP System
File name:	odtpfile
Collection interval:	Based on the table collection interval
Default retention:	1 day

Data Item	Type	Description
Time	Double	The time this information was recorded in the database for this sample
Interval	Double	The desired sample interval in this collection period
Svc_ReqQ	Double	The current number of service requests queued in this sample interval
Config_ReqQ	Double	The maximum configured request queue limit
Servers_Actv	Double	The current number of active servers doing work
Servers_Wait	Double	The current number of servers waiting for new work
Servers_Avail	Double	The current number of servers available and waiting for additional work for the same transaction
Servers_Attch	Double	The current number of servers currently attached

Servers_Max	Double	The maximum number of servers attached since the transaction management system control (TMSC) processor started
Clients_Attch	Double	The current number of clients currently attached
Clients_Max	Double	The maximum number of clients attached since the transaction management system control (TMSC) processor started
Threads_Curr	Double	The current number of active threads
Threads_Max	Double	The maximum number of active threads since the transaction management system control (TMSC) processor started
Threads_Resv	Double	The current number of threads reserved for servers that are not active
Threads_Config	Double	The maximum number of configured threads

A.10.2. Server Table Information

The following information is stored in the performance database tables:

Table Field Hierarchy

Class:	2200
Name:	ODTP Server
File name:	odtpfile
Collection interval:	Based on the table collection interval
Default retention:	1 day

Data Item	Type	Description
Time	Double	The time this information was recorded in the database for this sample
Interval	Double	The desired sample interval in this collection period
Server	String	The name of the server
Type	String	The server type (TIP, Batch, TIP-Sched)
ReqQ	Double	The number of requests queued in this sample interval
Fails	Double	The number of failures in this sample interval
Aborts	Double	The number of aborts in this sample interval
Sleeps	Double	The number of sleeps in this sample interval
Sleep_Time	Double	The sleep time in this sample interval
Receives	Double	The number of receives in this sample interval
Sends	Double	The number of sends in this sample interval

N2PC_Calls	Double	The number of service calls issued to other services that did not use two-phase commit in this sample interval
N2PC_Reqs	Double	The number of service requests processed that were not within a two-phase commit in this sample interval
2PC_Calls	Double	The number of service calls issued to other services that used two-phase commit in this sample interval
2PC_Reqs	Double	The number of service requests processed that were within a two-phase commit in this sample interval
Count	Integer	The number of combined occurrences. This is collected when STATS_LEVEL >= 2.
2PC_Txns	Double	Number of two-phase commit transactions initiated by a service in this sample interval
2PC_Time/Txn	Double	Average time for each two-phase commit service transactions in this sample interval
Txn_Commits	Double	Number of transactions committed that were initiated by a service in this sample interval
Txn_Rollbacks	Double	Number of transactions rolled back that were initiated by a service in this sample interval
Txn_Timeouts	Double	Number of transactions that timed out that were initiated by a service in this sample interval

A.10.3. Client Table Information

The following information is stored in the performance database tables:

Table Field Hierarchy

Class:	2200
Name:	ODTP Client
File name:	odtpfile
Collection interval:	Based on the table collection interval
Default retention:	1 day

Data Item	Type	Description
Time	Double	The time this information was recorded in the database for this sample
Interval	Double	The desired sample interval in this collection period
Client	String	The name of the client which is the tp_apname specified in the AP$NAME element of the client program
Gen_Runid	String	The generated OS 2200 run-id

A.10.4. Service Table Information

The following information is stored in the performance database tables:

Table Field Hierarchy

Class:	2200
Name:	ODTP Service
File name:	odtpfile
Collection interval:	Based on the table collection interval
Default retention:	1 day

Data Item	Type	Description
Time	Double	The time this information was recorded in the database for this sample
Interval	Double	The desired sample interval in this collection period
Service	String	The name of this service
Type	String	The service type (Local, Remote, HAA, or TWRAP)
Server	String	The configured name of the server or TRANCODE for an HAA service or TWRAP service

Fails	Double	The number of service requests that failed in this sample interval
Aborts	Double	The number of service requests processed that aborted in this sample interval
Sleeps	Double	The number of times waiting for messages from a a call or a conversation in this sample interval
Sleep_Time	Double	The time spent waiting for messages from a call or a conversation in this sample interval
Receives	Double	The number of conversational receives issued in this sample interval
Sends	Double	The number of conversational sends issued in this sample interval
N2PC_Calls	Double	The number of service calls issued to other services that did not use two-phase commit in this sample interval
N2PC_Reqs	Double	The number of service requests processed that were not within a two-phase commit in this sample interval
2PC_Calls	Double	The number of service calls issued to other services that used two-phase commit in this sample interval
2PC_Reqs	Double	The number of service requests processed that were within a two-phase commit in this sample interval
Count	Integer	The number of combined occurrences. This is collected when STATS_LEVEL >= 2.
Txn_Commits	Double	The number of transactions committed that were initiated by this service in this sample interval
Txn_Rollbacks	Double	The number of transactions rolled back that were initiated by this service in this sample interval
Txn_Timeouts	Double	The number of transactions that timed out that were initiated by this service in this sample interval

Appendix B
Predefined Report Descriptions

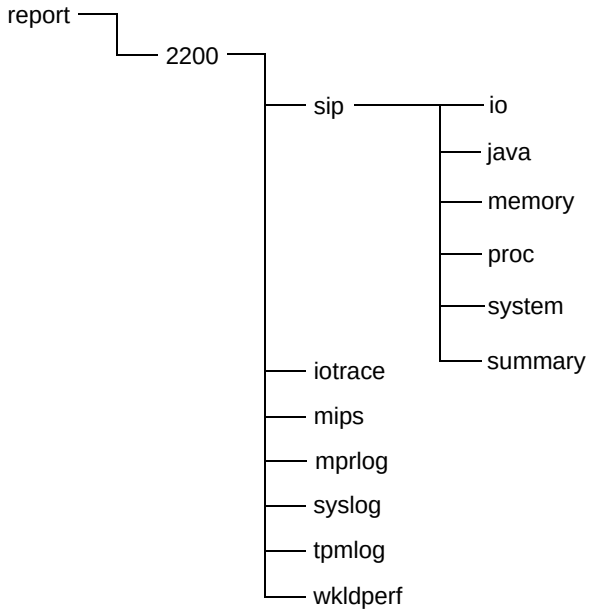
B.1. Introduction

Each of the probes released with the TeamQuest Performance Software includes a set of predefined reports. With TeamQuest View, you view these reports with data from the performance database. The reports are defined in the report definition files. The definition file determines the parameters displayed in the report, and in addition, the format and layout of the report. During a TeamQuest View session, you can change the view type and layout of a report using the menu options and commands available through TeamQuest View.

This appendix provides a list of the predefined reports for your 2200 host system and a brief description of the contents of each report.

B.2. Report Definition Directory

The report definitions are copied into a directory named report during installation. The report directory is organized into a hierarchy of subdirectories. The subdirectory structure is organized as follows:



The report directory contains a subdirectory for the system, in this case the 2200. The 2200 subdirectory contains a subdirectory for each available probe. The sip subdirectory contains subdirectories for input/output (io), memory, processor (proc), system, and summary reports. The iotrace subdirectory contains IOTRACE reports. The mips subdirectory contains MIPS probe reports. The mprlog subdirectory contains MAPPER log reports. The syslog subdirectory contains system log reports. The tpmlog subdirectory contains TMP log reports. The wkldperf subdirectory contains reports for workload service levels and facility utilizations.

Note: *If you are using TeamQuest Online, only report definitions in the SIP set of directories will have meaningful results.*

B.3. SIP Reports

The SIP reports described in this subsection provide detailed information about selected areas of system performance.

B.3.1. I/O Reports

The I/O report definitions are found in the io subdirectory of the sip subdirectory.

File:	CU-UNIT.RPT
Name:	Activity by Control Unit
Description:	Gives detailed statistics for the selected control units.

File:	CU-TOPN.RPT
Name:	Most Active Control Units
Description:	Gives detailed statistics for the top *n* control units.

File:	CD-DKPER.RPT
Name:	Cache Disk Percent
Description:	Provides a summary of cache actions.

File:	RLP-RATE.RPT
Name:	RLP Rates
Description:	Shows record lock processor (RLP) rates.

File:	MH-REQ.RPT
Name:	Multihost Reqs/Sec
Description:	Provides system-wide requests per second information on Multihost usage. Parameters presented are Lock Reqs/Sec, Unlock Reqs/Sec, Delayed Resp Msg/Sec, and Deadlocks/Sec.

File: MHTIPREQ.RPT

Name: Multihost TIP Reqs/Sec

Description: Provides total TIP requests per second information on Multihost usage. Parameters presented are Lock Reqs/Sec, Unlock Reqs/Sec, Delayed Resp Msg/Sec, and Deadlocks/Sec.

File: MHUDSREQ.RPT

Name: Multihost UDS Reqs/Sec

Description: Provides total UDS requests per second information on Multihost usage. Parameters presented are Lock Reqs/Sec, Unlock Reqs/Sec, Delayed Resp Msg/Sec, and Deadlocks/Sec.

File: MH-TOT.RPT

Name: Multihost Reqs

Description: Provides system-wide total requests information on Multihost usage. Parameters presented are Lock Reqs, Unlock Reqs, Delayed Resp Msg, and Deadlocks.

File: MHTIPTOT.RPT

Name: Multihost TIP Reqs

Description: Provides total TIP requests information on Multihost usage. Parameters presented are Lock Reqs, Unlock Reqs, Delayed Resp Msg, and Deadlocks.

File: MHUDSTOT.RPT

Name: Multihost UDS Reqs

Description: Provides total UDS requests information on Multihost usage. Parameters presented are Lock Reqs, Unlock Reqs, Delayed Resp Msg, and Deadlocks.

File: LOCKTOT.RPT

Name: UDS Lock Counts

Description: Provides total counts on the various UDS RLP lock functions. These parameters are collected for XPCs when the UDSTRP SIP counter exists.

File: LOCKREQ.RPT

Name: UDS Lock Reqs/Sec

Description: Provides total counts per second for the various UDS RLP lock functions. These parameters are collected for XPCs when the UDSTRP SIP counter exists.

File: UNLKTOT.RPT
Name: UDS Unlock Counts
Description: Provides total counts on the various UDS RLP unlock functions. These parameters are collected for XPCs when the UDSTRP SIP counter exists.

File: UNLKREQ.RPT
Name: UDS Unlock Reqs/Sec
Description: Provides total counts per second for the various UDS RLP unlock functions. These parameters are collected for XPCs when the UDSTRP SIP counter exists.

File: MISCTOT.RPT
Name: UDS Misc Total
Description: Provides total counts on the various UDS RLP function requests other than lock and unlock. These parameters are collected for XPCs when the UDSTRP SIP counter exists.

File: MISCREQ.RPT
Name: UDS Misc Reqs/Sec
Description: Provides total counts per second for the various UDS RLP function request other than lock and unlock. These parameters are collected for XPCs when the UDSTRP SIP counter exists.

File: CD-LCRET.RPT
Name: Cache LC RET
Description: Gives request existence times for each of the selected logical channels broken down by cache action.

File: TAP-UNIT.RPT
Name: Activity by Tape Unit
Description: Provides detailed I/O information for each selected tape unit.

File: TAP-TOPN.RPT
Name: Most Active Tape Units
Description: Provides detailed I/O information for the top *n* tape units.

File: CD-DKRET.RPT
Name: Cache Disk RET
Description: Provides overall request existence time information for cache disks broken down by cache action.

File: LC-UNIT.RPT
Name: Activity by LC
Description: Shows detailed I/O information for the selected logical channels.

File: LC-TOPN.RPT
Name: Most Active Logical Channels
Description: Shows detailed I/O information for the top *n* logical channels.

File: DSK-UNIT.RPT
Name: Activity by Disk Unit
Description: Gives detailed I/O information for the selected disk units.

File: DSK-TOPN.RPT
Name: Most Active Disk Units
Description: Gives detailed I/O information for the top *n* disk units.

File: CD-SUM.RPT
Name: Cache Summary
Description: Contains overall cache summary information.

File: CD-LCPER.RPT
Name: Cache LC Percent
Description: Shows the percentage of requests handled by each cache action for the selected logical channels.

File: PC-UNIT.RPT
Name: Physical Chan Activity
Description: Shows detailed I/O information for the selected physical channels.

File: PC-TOPN.RPT
Name: Most Active Physical Channels
Description: Shows detailed I/O information for the top/bottom *n* physical channels.

File: IO-TOT.RPT
Name: Total IOs/Second
Description: Shows total system IOs per second as well as the minimum and maximum for the time range.

The following predefined report definitions provide for accessing the performance statistics gathered by the OS 2200 SIP probe for the XPC. The reports are in the io subdirectory of the sip subdirectory.

File:	XPC-RET.RPT
Name:	XPC Request Existence Time
Description:	Provides request existence time information for the selected XPC broken down by XPC function.

File:	XPC-REQ.RPT
Name:	XPC Requests/Second
Description:	Provides requests per second information for the selected XPC broken down by XPC function.

File:	XIIP-ACT.RPT
Name:	Activity by XIIP
Description:	Gives specific XIIP performance information.

B.3.2. Java Reports

The java report definitions are found in the java subdirectory of the sip subdirectory. Statistics in the following reports are available on systems where it is possible to designate a processor or set of processors as dedicated Java IPs.

File:	BY-UNIT.RPT
Name:	Java Proc Util by Unit
Description:	Provides detailed Java processor utilization information for the selected processors.

File:	INT-SUM.RPT
Name:	Java Hardware Int Sum
Description:	Summarizes Java hardware interrupts. This information is only available if you have the hardware monitor option.

File:	AVL-UNIT.RPT
Name:	Java Proc Avail by Unit
Description:	Provides Java processor availability information.

File:	AVL-SUM.RPT
Name:Java	Proc Avail Summar
Description:	Summarizes Java processor availability.

File: INT-UNIT.RPT
Name: Java Processor Int by Unit
Description: Gives hardware interrupts by Java processor. This information is only available if you have the hardware monitor option.

File: UTIL-SUM.RPT
Name: Java Processor Utilization
Description: Provides Java processor utilization statistics for user categories.

File: IDLE.RPT
Name: Java Processor Utilization Idle
Description: Provides Java processor utilization statistics for idle as well as the minimum and maximum idle for the time range.

File: UTIL-TIP.RPT
Name: Java Processor Utilization TIP
Description: Provides TIP Java processor utilization as well as the minimum and maximum values for the time range.

B.3.3. Memory Reports

The memory report definitions are found in the memory subdirectory of the sip subdirectory.

File: AVL-MEM.RPT
Name: Available Memory
Description: Gives a summary of available memory.

File: LD-INITB.RPT
Name: Initial Bank Load Rate
Description: Provides initial bank load rates broken down by the various executable categories.

File: LD-INITP.RPT
Name: Init Program Load Rate
Description: Provides initial program load rates broken down by the various executable categories.

File: LD-DYNB.RPT
Name: Dynamic Bank Load Rate
Description: Provides dynamic bank load rates broken down by the various executable categories.

File: PAGE-WS.RPT
Name: Paging Working Sets
Description: Provides paging working set information.

File: PAGE-ACT.RPT
Name: Paging Activity Summary
Description: Gives a summary of paging activity.

File: PAGE-FLT.RPT
Name: Paging Fault Activity
Description: Gives paging fault activity information.

File: REAL-USE.RPT
Name: Real Space Usage
Description: Gives real space usage for the various page states.

B.3.4. Processor Reports

The processor report definitions are found in the proc subdirectory of the sip subdirectory.

File: SW-QUEUE.RPT
Name: Switching Queues
Description: Provides information on the switching queue when SIP level 4 or higher is configured.

File: BY-UNIT.RPT
Name: Proc Util by Unit
Description: Provides detailed processor utilization information for the selected processors.

File: INT-SUM.RPT
Name: Hardware Int Sum
Description: Summarizes hardware interrupts. This information is only available if you have the hardware monitor option.

File: AVL-UNIT.RPT
Name: Proc Avail by Unit
Description: Provides processor availability information.

File: AVL-SUM.RPT
Name: Proc Avail Summar
Description: Summarizes processor availability.

File: INT-UNIT.RPT
Name: Processor Int by Unit
Description: Gives hardware interrupts by processor. This information is only available if you have the hardware monitor option.

File: UTIL-SUM.RPT
Name: Processor Utilization
Description: Provides processor utilization statistics for user categories.

File: SW-UNIT.RPT
Name: Proc Util by Unit
Description: Shows detailed processor utilization by switching level information for the selected processors. The statistics displayed on this report are created when SIP level 4 or higher is configured.

File: SW-SUM.RPT
Name: Processor Utilization
Description: Shows summary processor utilization by switching level information for the system. The statistics displayed on this report are created when SIP level 4 or higher is configured.

File: IDLE.RPT
Name: Processor Utilization Idle
Description: Provides processor utilization statistics for idle as well as the minimum and maximum idle for the time range.

File: UTIL-TIP.RPT
Name: Processor Utilization TIP
Description: Provides TIP processor utilization as well as the minimum and maximum values for the time range.

B.3.5. System Reports

The system report definitions are found in the system subdirectory of the sip subdirectory.

File: BAT-RUNS.RPT
Name: Batch Run Summary
Description: Summarizes batch population and response information.

File: CAL-TOPN.RPT
Name: "Most active CALLs Used"
Description: Shows the *n* most active Exec call interfaces that are being utilized on the system.

File: DEM-RESP.RPT
Name: Demand Response Summary
Description: Lists demand response times in various time ranges.

File: DEM-RUNS.RPT
Name: Demand Run Summary
Description: Summarizes demand population and response information.

File: ER-SUM.RPT
Name: ER & CALL Summary
Description: Provides Executive request (ER) and call gate counts.

File: ER-TOPN.RPT
Name: "Most active ERs Used"
Description: Shows the *n* most active ERs that are being utilized on the system.

File: SUMMARY.RPT
Name: System Summary
Description: Gives a general overview of the system. This report includes processor utilization, memory utilization, and I/O counts.

File: TIP-SUM.RPT
Name: TIP Summary
Description: Shows memory and processor utilization for TIP.

File: QUE-AVL.RPT
Name: Available Queue
Description: Reports available queue information such as enqueued rates, queue times, and expansion and contraction rates.

File: QUE-SUM-RPT
Name: Queue Summary
Description: Reports information on the number of queue banks, number of available queues, and the number of deactivations per second.

File: QUE-SYS.RPT
Name: System Return Queue
Description: Reports system return information including queued entries per second, time spent on queues, and the number of queue server activations.

File: TIP-TXNS.RPT
Name: TIP Txns/Second
Description: Shows TIP transactions per second as well as the minimum and maximum values for the time range.

B.3.6. Summary Reports

The summary reports provide statistics, which give a general overview of system performance. These reports are found in the summary subdirectory of the sip subdirectory.

File: PR-UTIL.RPT
Name: Processor Utilization
Description: Reports processor utilization for user categories.

File: DEM-RESP.RPT
Name: Demand Response
Description: Reports demand response time.

File: LC-CONCR.RPT
Name: LC Concurrency
Description: Reports logical channel concurrency for each channel selected.

File: DSK-ACT.RPT
Name: Disk Activity
Description: Reports disk information for each selected disk.

File: IO-SUM.RPT
Name: I/O Summary
Description: Summarizes I/O rates for the system.

File: LC-IOS.RPT
Name: Logical Channel I/Os
Description: Reports I/O counts for the selected logical channels.

File: OPEN-RUN.RPT
Name: Runs Open
Description: Reports demand and batch population information.

File: TIP-SUM.RPT
Name: TIP Summary
Description: Shows total TIP transaction schedule and completion information.

File: PC-ACT.RPT
Name: Physical Chan Activity
Description: Reports physical channel information for each logical channel selected.

File: LC-ACT.RPT
Name: LC Activity
Description: Reports logical channel information for each logical channel selected.

File: DSK-SUM.RPT
Name: Disk I/O Summary
Description: Reports disk summary statistics.

File: XPC-SUM.RPT
Name: XPC Summary
Description: Contains overall XPC performance information.

File: XIIP-SUM.RPT
Name: XIIP Summary
Description: Contains overall XIIP performance information.

File: MEM-WS.RPT
Name: Memory Working Sets
Description: Gives information about memory working sets.

File: PAGE-ACT.RPT
Name: Paging Activity
Description: Provides a summary of paging activity.

File: MEM-UTIL.RPT
Name: Memory Utilization
Description: Reports memory utilization information.

B.4. IOTRACE Reports

The IOTRACE reports described in this subsection provide information about the workloads defined for the system. The IOTRACE report definition files are found in the iotrace subdirectory.

Note: *These reports are not available for TeamQuest Online.*

File: IOREQS.RPT
Name IO Requests
Description: Gives statistics for each workload on the number of I/O requests, I/O requests that were reads, and I/O requests that were cached.

File: WRDXFER.RPT
Name: Words Transferred
Description: Gives statistics for each workload on the number of words per input request and the number of words per output request.

File: SVCTIME.RPT
Name: Service Time
Description: Gives statistics for each workload on the average time required to process the I/O and the average time required to transfer the data.

File: SUMMARY.RPT
Name: Summary Information
Description: Gives statistics for each workload on the number of I/O requests, I/O requests that were reads, I/O requests that were cached, number of words per input request, number of words per output request, the average time required to process the I/O, and the average time required to transfer the data.

B.5. MIPS Report

The MIPS report described in this subsection provides information about MIPS usage for a Capacity On Demand (COD) system. The MIPS report is found in the mips subdirectory.

Note: *This report is not available for TeamQuest Online.*

File:	MIPS.RPT
Name:	MIPS Usage
Description:	Reports MIPS usage information including average MIPS used, Baseline MIPS, Ceiling MIPS, Desired MIPS, and Running MIPS.

B.6. MAPPER Log Reports

The MAPPER log reports described in this subsection display statistics for the defined workloads and general statistics for the system. The following workload report definitions can be found in the mprlog subdirectory.

Note: *These reports are not available for TeamQuest Online.*

File:	ACTIVE.RPT
Name:	Active Time
Description:	Gives statistics for each workload on total function active time in milliseconds.

File:	BRKPTS.RPT
Name:	Breakpoints
Description:	Gives statistics for each workload on the number of breakpoints that occurred.

File:	CONNECT.RPT
Name:	Connect Time
Description:	Gives statistics for each workload on the total time in seconds that stations were connected.

File:	FUNCS.RPT
Name:	MAPPER Functions
Description:	Gives statistics for each workload on the number of run functions, display functions, logged functions, logged functions outside of runs, functions processed, logged functions under runs, and started functions that occurred under runs.

File: INTERACT.RPT
Name: MAPPER Interactions
Description: Gives statistics for each workload on the number of terminal interactions that occurred.

File: IO.RPT
Name: MAPPER I/Os
Description: Gives statistics for each workload on the number of input and output operations processed, number input and output operations from functions under runs, and the number of input operations from functions under runs.

File: LINES.RPT
Name: Lines Processed
Description: Gives statistics for each workload on the number of lines read and written from functions under runs.

File: RESPONSE.RPT
Name: MAPPER Response
Description: Gives the statistic for each workload on the number of seconds per response and the number of seconds per logged function.

B.7. System Log Reports

The system log reports described in this section provide information about the workloads defined for the system. The workload report definition files are found in the syslog subdirectory.

Note: *These reports are not available for TeamQuest Online.*

File: TOTSUPS.RPT
Name: Total SUPs
Description: Gives statistics for the selected workload and environment (total, TIP, demand, batch, or deadline) on total SUPs.

File: IPSUPS.RPT
Name: IP SUPs
Description: Gives statistics for the selected workload and environment (total, TIP, demand, batch, or deadline) on IP SUPs.

File: IOSUPS.RPT
Name: IO SUPs
Description: Gives statistics for the selected workload and environment (total, TIP, demand, batch, or deadline) on I/O SUPs.

File:	ERCCSUPS.RPT
Name:	ER/CC SUPs
Description:	Gives statistics for the selected workload and environment (total, TIP, demand, batch, or deadline) on ER and control card SUPs.

File:	ESCSSUPS.RPT
Name:	ES GCS SUPs
Description:	Gives statistics for the selected workload and environment (total, TIP, demand, batch, or deadline) on the Exec Services and control statement IO and IP SUPs.

File:	CBSUPS.RPT
Name:	Core Block SUPs
Description:	Gives statistics for the selected workload and environment (total, TIP, demand, batch, or deadline) on core block SUPs.

File:	SUPS.RPT
Name:	SUPs by Workload
Description:	Gives statistics for the selected workload and environment (total, TIP, demand, batch, or deadline) on IP SUPs, ER and control card SUPs, and IO SUPs.

File:	SVCLVL.RPT
Name:	Service Levels by Workload
Description:	Gives statistics for the selected workload and environment (total, TIP, demand, batch, or deadline) on total SUPs voluntary wait time, resource wait time, and active time.

File: THRUPUT.RPT

Name: Throughput by Workload

Description: Gives statistics for the selected workload and environment (total, TIP, demand, batch, or deadline) on number of I/O requests, disk I/O requests, disk I/O words transferred, tape I/O requests, tape I/O words transferred, number of transactions, cards the Exec read in, and the number of runs opened.

File: SUMMARY.RPT

Name: Information by Workload

Description: Gives statistics for the selected workload and environment (total, TIP, demand, batch, or deadline) on total SUPs, IP SUPs, ER and control card SUPs, IO SUPs, core block SUPs, voluntary wait time, resource wait time, active time, number of I/O requests, disk I/O requests, disk I/O words transferred, tape I/O requests, tape I/O words transferred, number of transactions, cards the Exec read in, and the number of runs opened, real pages concurrently resident, and real memory usage.

B.8. TPM Log Reports

The TPM reports provide information about the workloads defined for the system. The TPM log reports are found in the tpmlog subdirectory.

Note: *These reports are not available for TeamQuest Online.*

File: WORKLOAD.RPT

Name: Information by workload

Description: Gives statistics for each workload on total SUPs and transaction count.

File: OUTPUT.RPT

Name: Output Messages

Description: Gives statistics for each workload on output message number and size.

File: TXNRSP.RPT

Name: Transaction response time

Description: Gives statistics for each workload on total transaction response time.

File: TXNTIME.RPT

Name: Transaction time

Description: Gives statistics for each workload on the components of transaction time.

File:	FCSS.RPT
Name:	FCSS Function Requests
Description:	Gives statistics for each workload on the number of FCSS read, write, lock, and other requests.

File:	FCSSWORD.RPT
Name:	FCSS Words
Description:	Gives statistics for each workload on the total number of FCSS words read and written.

File:	LINC.RPT
Name:	LINC Read/Writes
Description:	Gives statistics for each workload on the total number of Exec and TIP read and write requests.

File:	LOCKS.RPT
Name:	Locks
Description:	Gives statistics for each workload on the number of locks, time waiting on locks, and the time locks were held.

File:	SUPS.RPT
Name:	SUPs by Workload
Description:	Give statistics for each workload on the total IP, ER/CC, and I/O SUPs.

File:	TXNCNT.RPT
Name:	Transaction count
Description:	Gives the statistics for each workload on the number of transactions, number of passoff messages, number of I/O requests, and response, processing, and termination average commit time.

File:	RTPS.RPT
Name:	RTPS counts
Description:	Gives statistics for each workload on the number of RTPs deactivated transactions and the number of RTPs deactivated copies with the same transaction code.

File:	TXNACT.RPT
Name:	Transactions active
Description:	Gives statistics for each workload on the number of stuck copies, total stuck, active copies, and total active counts.

B.9. Workload Performance Reports

The workload performance reports provide information about workload service levels and facility utilizations. These reports are found in the wkldperf subdirectory, and provide derived statistics information from the SIP (**tqbsp**), TPM log (**tqtpm**), MAPPER log (**tqmpr**), and system log (**tqsys**) probes. These reports are intended for top-down performance analysis activity that may include capacity planning.

Note: *If you open these reports and no data values are reported for any of the statistics, you may need to load the workload performance derived statistic definitions into your database (see A.8).*

File:	summary.rpt
Name:	Service Levels and Facility Usage Summary
Description:	Using statistics derived from the **tqbsp** and **tqsys** probes, this report shows overall average populations, throughputs, and response times for MAPPER, TIP, demand, deadline batch, and batch environments, respectively. The report also shows total CPU utilization, disk IOs per second, average disk IO response time, and memory utilization.

File:	allbatch.rpt
Name:	Overall Batch Performance
Description:	Using statistics derived from the **tqsys** probe, this report shows overall average batch environment population, throughput, response time, IP SUPs per second, IO SUPs per second, ER/CC SUPs per second, disk IOs per second, tape IOs per second, and memory usage per batch run.

File:	alldeadl.rpt
Name:	Overall Deadline Performance
Description:	Using statistics derived from the **tqsys** probe, this report shows overall average deadline batch environment population, throughput, response time, IP SUPs per second, IO SUPs per second, ER/CC SUPs per second, disk IOs per second, tape IOs per second, and memory usage per deadline batch run.

File:	alldmand.rpt
Name:	Overall Demand Performance
Description:	Using statistics derived from the **tqsys** probe, this report shows overall average demand environment population, throughput, response time, IP SUPs per second, IO SUPs per second, ER/CC SUPs per second, disk IOs per second, tape IOs per second, and memory usage per demand run.

File: alltip.rpt

Name: Overall TIP Performance

Description: Using statistics derived from the **tqsys** probe, this report shows overall average TIP environment population, throughput, response time, IP SUPs per second, IO SUPs per second, ER/CC SUPs per second, disk IOs per second, tape IOs per second, and memory usage per TIP transaction.

File: allhvtip.rpt

Name: Overall HVTIP Performance

Description: Using statistics derived from the **tqtpm** and **tqrtpm** probes, this report shows overall average HVTIP environment population, throughput, response time, CMS queue time, processing time, lock wait time, termination time, IP SUPs per second, ER/CC SUPs per second, IO SUPs per second, FCSS reads per second, and FCSS writes per second.

File: allmappr.rpt

Name: Overall MAPPER Performance

Description: Using statistics derived from the **tqmpr** probe, this report shows overall average MAPPER environment population, throughput, response time, lines processed per second, and IOs processed per second.

File: batch.rpt

Name: Batch Workload Performance

Description: Using statistics derived from the **tqsys** probe, this report shows by workload: batch population, throughput, response time, IP SUPs per second, IO SUPs per second, ER/CC SUPs per second, disk IOs per second, tape IOs per second, and memory usage per batch run.

File: deadline.rpt

Name: Deadline Workload Performance

Description: Using statistics derived from the **tqsys** probe, this report shows by workload: deadline batch population, throughput, response time, IP SUPs per second, IO SUPs per second, ER/CC SUPs per second, disk IOs per second, tape IOs per second, and memory usage per deadline batch run.

File: demand.rpt

Name: Demand Workload Performance

Description: Using statistics derived from the **tqsys** probe, this report shows by workload: demand population, throughput, response time, IP SUPs per second, IO SUPs per second, ER/CC SUPs per second, disk IOs per second, tape IOs per second, and memory usage per demand run.

File:	tip.rpt
Name:	TIP Workload Performance
Description:	Using statistics derived from the **tqsys** probe, this report shows by workload: TIP population, throughput, response time, IP SUPs per second, IO SUPs per second, ER/CC SUPs per second, disk IOs per second, tape IOs per second, and memory usage per TIP transaction.

File:	hvtip.rpt
Name:	HVTIP Workload Performance
Description:	Using statistics derived from the **tqtpm** and **tqrtpm** probes, this report shows by workload: HVTIP population, throughput, response time, CMS queue time, processing time, lock wait time, termination time, IP SUPs per second, ER/CC SUPs/sec, IO SUPs/sec, FCSS reads per second, and FCSS writes per second.

File:	mapper.rpt
Name:	MAPPER Workload Performance
Description:	Using statistics derived from the **tqmpr** probe, this report shows by workload: MAPPER population, throughput, response time, lines processed per second, and IOs processed per second.

Appendix C
Report Definition File

C.1. Introduction

Each predefined report is defined in its own report definition file. You can also define your own reports to meet the needs of your site.

Use the information in this appendix to customize reports for your site. You can create new reports by either creating a new report in TeamQuest View and adding and deleting parameters or by creating your own reports using a text editor and entering the file format described in this appendix.

The following example shows a report definition file. The name of the report definition file is dsk-unit.rpt and it defines the Activity by Disk Unit report.

```
[general]
Copyright = Copyright (c) 2012 TeamQuest Corporation. All Rights Reserved.
Host Type = "2200"
Report Name = "Activity by Disk Unit"
Format = Date+Time,
      %System%+%Application Group%+%Workload Set%+%Workload%+%Category
      Group%+%Category%+%Subcategory%+%Statistic%+%Value%,
      %Resource%

[Parm0]
System = "[?]"
Category Group = "IO Activity"
Category = "Disk"
Subcategory = "Summary"
Statistic = "Reqs/Sec", "KWords/Req", "Percent Busy",
      "RET-FLD", "Hardware Serv Time", "Transfer Time",
      "Concurrency"
Resource = [?]
Value Types = Average
```

C.2. Notation Conventions

The following conventions apply to the format of a report definition file as described in this appendix:

- The text that appears in normal type must be entered exactly as shown. However, you can use either uppercase or lowercase characters.
- Italics indicate fields or values that you must supply.
- Multiple choices are separated by commas. You must type the comma between fields.
- Parentheses ( ) indicate optional choices. Do not enter the parentheses.
- Some terms are enclosed in braces ({ }) and separated by a vertical bar (|). You must choose one of the terms enclosed within the braces. Do not type the braces.
- Section headers must be enclosed within brackets ([ ]) within the report definition.
- The comma (,) is the continuation character. If the syntax does fit on a single line, complete the field, end the line with a comma, and continue on the next line.
- For any field, leading and trailing spaces are ignored. If a field contains spaces, it must be enclosed in quotation marks (" ").
- Fields enclosed in quotation marks cannot contain commas.

C.3. File Format

A report definition file is divided into sections with each section defining a specific aspect of the report. Each section contains a section header followed by the definition items that provide the specifications for the report. The sections for a report definition file are

- General—this section defines the attributes for the layout and object attributes of the report. TeamQuest View uses the specifications in this section for the initial presentation of a report.
- Parameter-Group—these sections define the data series that are included in the report. One parameter-group section appears for each data series in the report.

C.4. General Section

The general section contains the specifications for the layout of the report.

Format

The format for the general section is

```
[general]
Copyright = Copyright (c) 2012 TeamQuest Corporation. All Rights Reserved.
Product Name = text
Report Name = text
Host Type = host-type
Date Format = format
Time Format = format
Aggregation Set = aggregation-set-name
Report Type = {Monitor|History}
Date Multiplier = positive-integer
Point Size = positive-integer
Time = start, end
Date = {CalendarType, }RelativeDate(, RelativeDate ...)
View = {Spread|Sbar|Pie|Cluster|Line|Area|Point|Scatter|Trend}
Format = XAxis, LegendEntries(, Page)
Trend = Series, Line, Order, Equation
Scatter = XSeries, YSeries, Line, Order, Equation
XAxis Title = text
Value Title = text
Title Style = Font, Style, Size, Foreground, Background,Border
Value Style = Font, Style, Size, Foreground, Background, Border
Value Axis = Enable, Font, Style, Size, Color
Value Line = Enable, Thick, MajorTick, MinorTick, MajorGrid,
       MinorGrid
Values = Major, Minor, Minimum, Maximum(,Metric)
XAxis Style = Font, Style, Size, Foreground, Background, Border
XAxis = Enable, Font, Style, Size, Color
XAxis Line = Enable, Thick, MajorTick, MinorTick, MajorGrid,
       MinorGrid
XAxis Labeling = Major, Minor, Midnight
Point Consolidation = Enable, Points
Legend = Enable, Location(, top, left, bottom, right)
Legend Style = Font, Style, Size, Foreground, Background, Border
Threshold Color = Normal, Exception
Threshold Pattern = Normal, Exception
Window Area = top, left, bottom, right
Background Color = Color
Auto Multiplier = {ON|OFF}
Display Markers = {ON|OFF}
BFNote = Top, Left, Bottom, Right, Vertical, Horizontal, Font,
       Style, Size, Foreground, Background, Border
Note = Text, Top, Left, Bottom, Right, Vertical, Horizontal,
       Font, Style, Size, Foreground, Background, Border
Export Data = Page, Spread, Comma
Sequence = Color, Hatch, Marker, Line
```

The following descriptions explain the syntax of the general specifications of the report definition:

Copyright = Anything You Want!
: The copyright statement is never saved by TeamQuest View. This statement is used by TeamQuest View when it reads this definition to identify that it is not to overwrite the file when you use the File > Save menu selection. You can still overwrite this definition by using the File > Save As menu selection and specifying the same file name. When you save the report this way, the copyright statement is not written to the file.

Product Name = *text*
: The name of the product for which this report is available.

  text
  : The name of a TeamQuest Performance Software product to which the report definition applies.

Report Name = *text*
: The name of the report as it appears on the title bar of the report window.

  text
  : Can be a string of characters up to 50 characters long. If it contains spaces, it must be enclosed in quotation marks (" ").

Host Type = *host-type*
: The type of computer hardware with which this report can be used. Each hardware type has a unique set of host-defined key names. The combination of keys is unique for a particular class of hardware. This specification must equal the host type provided by the host product. If this specification is made, a check is done with the host as identified by the host product. If a mismatch occurs, the user can ignore the error or cancel.

Date Format = *format*
: The specifications for the format of the date as it appears on the report.

  format
  : Can be any of the following and must be entered in uppercase:

YY	Years (00–99)
YYYY	Years (specified with the century)
MM	Month (01–12)
MMM	Month (abbreviated name)
MMMM	Month (complete name)
DD	Day of the month (01–31)
DDD	Day of the year (001–366)
DDDD	Day of the week (0–6, where 0 = Sunday)
WW	Week of the year (0–53)

WWW	Day of the week (abbreviated name)
WWWW	Day of the week (complete name)

The default values for Date are defined in the Date/Time Formats settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

Time Format = *format*
The specifications for the format of the time as it appears on the report.

format
Can be any of the following and must be entered in lowercase:

hh	Hours (00–23)
hhh	Hours (00–12)
mm	Minutes (00–59)
ss	Seconds (00–59)
pp	AM or PM indicator

The default values for Time are defined in the Date/Time Formats settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

Aggregation Set = *aggregation-set-name*
The name for the aggregation set in the database to use for data retrieval.

Report Type = {Monitor | History}
The type of report for the report definition.

You can specify either Monitor or History.

Monitor
Indicates that this report is used for real-time monitoring.

History
Indicates that this report is used for reporting historical data.

If this field is not present, the report defaults to the calendar from the session.

Date Multiplier = *positive-integer*
The number used to consolidate data across time groups in the aggregation set. For example, if daily data is stored in the database and a date multiplier of 7 is specified, the report shows data consolidated into one sample for every seven days of data. The specification applies to history reports only.

positive-integer
Is a positive integer that is used as the date multiplier. The default value is 1. A value less than 2 causes no consolidation.

Point Size = *size*

Specifies the size of one data point in seconds. *Size* can be any real integer. If Point Size = 3600, the granularity of the data in the aggregation set defines the time span of the data points graphed. For example, if the granularity of the data in the aggregation set is 30 seconds, then 30-second data points will be retrieved. A value of 3600 indicates one point per hour. If the granularity of the data is 30 minutes, then two data points are consolidated to create a single point.

Time = *start, end*

The time interval for which data is requested from the host. These values are expressed in a 24-hour clock notation.

start

Is the starting time for the time interval in the format *hh:mm:ss*.

end

Is the ending time for the time interval in the format *hh:mm:ss*.

The time interval cannot be greater than 24 hours. If the ending time is less than or equal to the starting time, TeamQuest View assumes that the time interval extends across midnight to the next day. If the report is a monitor, the difference between the start time and the end time determines the size of the report that is displayed.

Date = {*Calendar Type*,|*Relative Date*}(, *Relative Date* ...)

The relative date for the default calendar.

CalendarType

Is the Day, Week, Month, Quarter, or Year that determines the relative calendar type and the representation of *RelativeDate*. If *CalendarType* is not present, *CalendarType* is assumed to be Day. By definition, a day ends at 24:00:00 and a week ends on Saturday at 24:00:00.

RelativeDate

Is a relative integer which represents a *CalendarType* relative to the current date. If *CalendarType* is Day, 0 represents today, 1 represents yesterday, 2 represents the day before yesterday, and so on. If the *CalendarType* is Week, 0 represents the current week, 1 represents last week, and so on.

View = {Spread | Sbar | Pie | Cluster | Line | Area | Point | Scatter | Trend}

The type of graph or text report that TeamQuest View displays for the report. Only one type of report can be specified in this field. The default value for this field is Line. The following types of reports can be specified:

Spread	Displays data as textual reports in spreadsheet format.
Sbar	Displays data in stacked bar graph format.
Pie	Displays data in pie chart format.
Cluster	Displays data in cluster bar graph format.
Line	Displays data in line graph format.

Area	Displays data in area graph format.
Point	Displays data as points only.
Scatter	Displays data in scatter plot format. The data series and line type settings for the x- axis and y-axis are defined in the Scatter key.
Trend	Displays data in a bar graph format with a trend line. The series and line types are defined in the Trend key.

Format = *XAxis, LegendEntries(, Page)*

The specifications for the grouping of data along the axis. This determines how line, cluster bar, stacked bar, and scatter graphs appear.

XAxis

Determines the values placed along the horizontal axis of the report.

LegendEntries

Determines the source of the data series. Usually this field contains *parameter*.

Page

Determines how to separate and sort data for additional pages of a report.

The following values can be used for the *XAxis, LegendEntries,* and *Page* fields: *parameter*, Date, Time, and *%keyname%*.

A special *%keyname%* of *%value%* represents the Value Type field of a parameter group.

You cannot mix the specifications of *parameter* and *%keyname%*.

You can concatenate the specifications by using a plus (+) sign. Concatenated specifications are sorted by the order that the keywords are defined and then by date and time.

The values contained in the *XAxis, LegendEntries,* and *Page* fields must not contain the same keys.

If the date and time specifications are not used in any of the fields, they are appended to the end of the *Page* field.

If any unused *%keyname%* values are present, they are appended to the *Page* field.

For example, in the following Format statement:

Format = %Resource%+%Statistic%, Time, Date

The data for the x-axis is sorted by resource and statistic. The legend is grouped according to time. The pages of the report are ordered by date.

Trend = *Series, Line, Order, Equation*

The specifications for the creation of a trend graph. This determines which data series is graphed and which type of trend line is displayed.

Series

The number of the data series that is graphed. The number is determined by its position displayed in the series list in the Trend Graph Setup dialog box. The top most series is number 1.

Line

Specifies the type of trend line. Valid trend line types are

Linear	Straight best fit line.
Polynomial	A trend polynomial curve of any order from 2 through 5. Order is determined by the order value.

Order

The order of the polynomial when using the polynomial line type. Valid values are: 2, 3, 4, or 5.

Equation

The setting to determine if the best fit line equation is displayed. The text attributes are defined in the BFNote key. Valid values are

ON	Display the best fit equation.
OFF	Do not display the best fit equation.

Scatter = *XSeries, YSeries, Line, Order, Equation*

The specifications for the creation of a scatter graph. This determines which data series are graphed and which type of best fit line is displayed.

XSeries

The number of the data series that was used along the x-axis. The number is determined by its position displayed in the x-axis list in the Scatter Plot Setup dialog box. The top most series is number 1.

YSeries

The number of data series that was used along the y-axis. The number is determined by its position displayed in the y-axis list in the Scatter Plot Setup dialog box. The top most series is number 1.

Line

Specifies the type of best fit line. Valid best fit line types are

None	No best fit line.
Linear	A linear best fit line.
Polynomial	A best fit polynomial curve of any order form 2 through 5. Order is determined by the order value.

Order
The order of the polynomial when using the polynomial line type. Valid values are 2, 3, 4, or 5.

Equation
The setting to determine if the best fit line equation is displayed. The text attributes are defined in the BFNote key. Valid values are

ON	Display the best fit equation.
OFF	Do not display the best fit equation.

XAxis Title = *text*
The title for the horizontal axis of the report. It appears under the x-axis of the report.

text
Can be a string of text up to 80 characters long. If it contains spaces, the title must be enclosed in quotation marks (" ").

Value Title = *text*
The title for the vertical axis of the report. It appears along the value axis of the report.

text
Can be a string of text up to 80 characters long. If the title contains spaces, it must be enclosed in quotation marks (" ").

Title Style = *Font, Style, Size, Foreground, Background, Border*
The specifications for the text attributes of the page title of the report.

Font
Specifies the typeface in which the page title is printed on the report.

Style
Specifies the kind of emphasis for the typeface of the page title. Some acceptable values are

Normal	For no emphasis
Bold	For highlighting the text
Italic	For emphasizing the text

You can specify multiple styles for the text. However, the styles must be joined by a plus (+) sign. For example, to specify that the title be printed in bold and italic typeface, you enter bold+italic in the Style field of the specification.

Size
Defines the number of points for the typeface of the page title. This value determines the height of the characters in which the page title is printed. The value must be a positive integer.

Foreground
Indicates the color in which the page title of the report is printed.

Background
Indicates the color of the area that appears directly behind the page title of the report. The background area is a boxed area and acts as a frame for the page title of the report. If a background color is specified, the boxed area behind the page title covers any graph lines and other text that may appear behind the page title. If the *Background* field for the page title is not specified, TeamQuest View allows any graph lines or other text that may be behind the page title to show through.

Border
Indicates the color of the border line that appears for the background area of the page title. If the *Border* field is not specified, TeamQuest View does not display a border line around the background area of the page title.

The default values for Title Style are defined in the Report Styles settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

Value Style = *Font, Style, Size, Foreground, Background, Border*
The specifications for the text attributes of the title of the vertical axis.

Font
Specifies the typeface in which the value title is printed.

Style
Specifies the kind of emphasis for the typeface of the value title. Some acceptable values are

Normal	For no emphasis
Bold	For highlighting the text
Italic	For emphasizing the text

You can specify multiple styles for the text. However, the styles must be joined by a plus (+) sign. For example, to specify that the title be printed in bold and italic typeface, you enter bold+italic in the Style field of the specification.

Size
Defines the number of points for the typeface of the value title. This value determines the height of the characters in which the value title is printed. The value must be a positive integer.

Foreground
Indicates the color in which the value title is printed.

Background
Indicates the color of the area that appears directly behind the value title of the axis. The background area is a boxed area and acts as a frame for the value title. If a background color is specified, the boxed area behind the value title covers any graph lines and other text that may appear behind the value title. If the *Background* field for the value title is not specified, TeamQuest View allows any graph lines or other text that may be behind the value title to show through.

Border
Indicates the color of the border line that appears for the background area of the value title. If the *Border* field is not specified, TeamQuest View does not display a border line around the background area of the value title.

The default values for Value Style are defined in Report Styles settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

Value Axis = *Enable, Font, Style, Size, Color*
The specifications for the text attributes of the labels for the vertical axis. The axis labels appear along the axis at the major tick marks and are used to identify the values of the vertical axis.

Enable
Indicates whether the labels appear along the vertical axis. The acceptable values are ON or OFF. If *Enable* is set to ON, the labels appear on the report. If *Enable* is set to OFF, the labels do not appear on the report. The default for this field is ON.

Font
Specifies the typeface in which the labels are printed.

Style
Specifies the kind of emphasis for the typeface of the labels. Some acceptable values are

Normal	For no emphasis
Bold	For highlighting the text
Italic	For emphasizing the text

You can specify multiple styles for the labels. However, the styles must be joined by a plus (+) sign. For example, to specify that the labels be printed in bold and italic typeface, you enter bold+italic in the Style field of the specification.

Size
Defines the number of points for the typeface of the labels. This value determines the height of the characters in which the labels are printed. The value must be a positive integer.

Color
Specifies the color in which the vertical axis and the labels are printed. The default value is black.

The default values for Value Axis are defined in the Axis Defaults settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

Value Line = *Enable, Thick, MajorTick, MinorTick, MajorGrid, MinorGrid*
The specifications for the line attributes of the vertical axis of the report.

Enable
Indicates whether the vertical axis line appears on the report. The acceptable values are ON or OFF. If *Enable* is set to ON, the vertical axis line appears on the report. If *Enable* is set to OFF, the vertical axis does not appear on the report. The default for this field is ON.

Thick
Defines the weight in pixels of the vertical axis line. The value for this specification must be a positive integer. The acceptable values are 1, 2, and 3. The default value is 1.

MajorTick
Defines the type of marks used for the major ticks on the vertical axis line. The acceptable values are

Invisible	No ticks marks appear.
Inside	Tick marks appear on the inside of the axis line.
Outside	Tick marks appear on the outside of the axis line.
Crossing	Tick marks cross the axis line.

The default value for the MajorTick field is crossing.

MinorTick
Defines the type of marks used for the minor ticks on the vertical axis line. The acceptable values are

Invisible	No ticks marks appear.
Inside	Tick marks appear on the inside of the axis line.
Outside	Tick marks appear on the outside of the axis line.
Crossing	Tick marks cross the axis line.

The default value for the MinorTick field is invisible.

MajorGrid

Indicates whether grid lines are drawn across the graph at the points of the vertical axis designated by the major ticks. The acceptable values are ON or OFF. If *MajorGrid* is set to ON, the grid lines appear on the report. If *MajorGrid* is set to OFF, the grid lines do not appear on the report. The default value is OFF.

MinorGrid

Indicates whether grid lines are drawn across the graph at the points of the vertical axis designated by the minor ticks. The acceptable values are ON or OFF. If *MinorGrid* is set to ON, the grid lines appear on the report. If *MinorGrid* is set to OFF, the grid lines do not appear on the report. The default value is OFF.

The default values for Value Line are defined in the Axis Defaults settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

Values = *Major, Minor, Minimum, Maximum*(,*Metric*)

The specifications for the presentation of parameter values on the report. The Values specifications determine the minimum and maximum values that are shown along the vertical axis and the distance between the major and minor tick marks on the axis.

Major

Is a real number representing the distance from the intersection of the vertical axis and the horizontal axis to the first major tick mark. Major tick marks appear on every integer multiple of the real number specified in this field. When you specify zero in this field, the value is determined automatically.

Minor

Is a real number representing the distance from the intersection of the vertical axis and the horizontal axis to the first minor tick mark. Minor tick marks appear on every integer multiple of the real number specified in this field. When you specify zero in this field, the value is determined automatically.

Major tick marks always override minor tick marks. The value specified in the Major field should be an integer multiple of the value specified in the Minor field.

Minimum

Is a real number representing the point where the vertical axis crosses the horizontal axis. Values less than the specified minimum value do not appear on the report. When you specify zero in this field, the value is determined automatically.

Maximum
Is a real number representing the point where the vertical axis ends on the report display. Values greater than the specified maximum value do not appear on the report. When you specify zero in this field, the value is determined automatically.

(, *Metric*)
Can be either ON or OFF.

ON	Displays a single character metric prefix character instead of scientific notation when the string representation of values exceeds 1000. Numbers outside the range of 1e27 and 1e–27 are still displayed as scientific notation.
OFF	Displays scientific notation when the size of the displayed number exceeds six digits. This is the default if not specified.

Ideally, the difference between the values specified in the minimum and the maximum fields should be an integer multiple of the value specified in the *Major* field.

If values are not specified for the *Minimum* and *Maximum* fields, TeamQuest View uses the autoscale feature to present parameter values.

XAxis Style = *Font, Style, Size, Foreground, Background, Border*
The specifications for the text attributes of the title of the horizontal axis.

Font
Specifies the typeface in which the x-axis title is printed.

Style
Specifies the kind of emphasis for the typeface of the x-axis title. Some acceptable values are

Normal	For no emphasis
Bold	For highlighting the text
Italic	For emphasizing the text

You can specify multiple styles for the text. However, the styles must be joined by a plus (+) sign. For example, to specify that the x-axis title be printed in bold and italic typeface, you enter bold+italic in the Style field of the specification.

Size
Defines the number of points for the typeface of the x-axis title. This value determines the height of the characters in which the x-axis title is printed. The value must be a positive integer.

Foreground
Indicates the color in which the x-axis title is printed.

Background
Indicates the color of the area that appears directly behind the x-axis title of the axis. The background area is a boxed area and acts as a frame for the x-axis title. If a background color is specified, the boxed area behind the x-axis title covers any graph lines and other text that may appear behind the x-axis title. If the *Background* field for the x-axis title is not specified, TeamQuest View allows any graph lines or other text that may be behind the x-axis title to show through.

Border
Indicates the color of the border line that appears for the background area of the x-axis title. If the *Border* field is not specified, TeamQuest View does not display a border line around the background area of the x-axis title.

The default values for XAxis Style are defined in the Report Styles settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

XAxis Line = *Enable, Thick, MajorTick, MinorTick, MajorGrid, MinorGrid*
The specifications for the line attributes of the horizontal axis of the report.

Enable
Indicates whether the horizontal axis line appears on the report. The acceptable values are ON or OFF. If *Enable* is set to ON, the horizontal axis line appears on the report. If *Enable* is set to OFF, the horizontal axis does not appear on the report. The default for this field is ON.

Thick
Defines the weight in pixels of the horizontal axis line. The value for this specification must be a positive integer. The acceptable values are 1, 2, and 3. The default value is 1.

MajorTick
Defines the type of marks used for the major ticks on the horizontal axis line. The acceptable values are

Invisible	No tick marks appear.
Inside	Tick marks appear on the inside of the axis line.
Outside	Tick marks appear on the outside of the axis line.
Crossing	Tick marks cross the axis line.

The default value for the MajorTick field is crossing.

MinorTick
Defines the type of marks used for the minor ticks on the horizontal axis line. The acceptable values are

Invisible	No tick marks appear.
Inside	Tick marks appear on the inside of the axis line.
Outside	Tick marks appear on the outside of the axis line.
Crossing	Tick marks cross the axis line.

The default value for the MinorTick field is invisible.

MajorGrid
Indicates whether grid lines are drawn across the graph at the points of the horizontal axis designated by the major ticks. The acceptable values are ON or OFF. If *MajorGrid* is set to ON, the grid lines appear on the report. If *MajorGrid* is set to OFF, the grid lines do not appear on the report. The default value is OFF.

MinorGrid
Indicates whether grid lines are drawn across the graph at the points of the horizontal axis designated by the minor ticks. The acceptable values are ON or OFF. If *MinorGrid* is set to ON, the grid lines appear on the report. If *MinorGrid* is set to OFF, the grid lines do not appear on the report. The default value is OFF.

The default values for XAxis Line are defined in the Axis Defaults settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

XAxis = *Enable, Font, Style, Size, Color*
The specifications for the text attributes of the labels for the horizontal axis. The axis labels appear along the axis at the major tick marks and are used to identify the values of the horizontal axis.

Enable
Exists in the syntax but does not affect the axis. The functionality of this field has been replaced by the x-axis labeling definition statement.

Font
Specifies the typeface in which the labels are printed.

Style
Specifies the kind of emphasis for the typeface of the labels. Some acceptable values are

Normal	For no emphasis
Bold	For highlighting the text
Italic	For emphasizing the text

You can specify multiple styles for the labels. However, the styles must be joined by a plus (+) sign. For example, to specify that the labels be printed in bold and italic typeface, you enter bold+italic in the Style field of the specification.

Size
Defines the number of points for the typeface of the labels. This value determines the height of the characters in which the labels are printed. The value must be a positive integer.

Color
Specifies the color in which the horizontal axis and the labels are printed. The default value is Black.

The default values for XAxis are defined in the Axis Defaults settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

XAxis Labeling = *Major, Minor, Midnight*
The specifications for the labels of the horizontal axis.

Major
Determines if labels appear at the major tick along the horizontal axis. The acceptable values are ON or OFF. If *Major* is set to ON, TeamQuest View displays labels at the major ticks. If *Major* is set to OFF, TeamQuest View does not display labels at the major ticks.

Minor
Determines if labels appear at the minor ticks along the horizontal axis. The acceptable values are ON or OFF. If *Minor* is set to ON, TeamQuest View displays labels at the minor ticks. If *Minor* is set to OFF, TeamQuest View does not display labels at the minor ticks.

Midnight
Determines if a label appears when the time interval crosses midnight. Acceptable values are ON or OFF. If *Midnight* is set to ON, a date label appears when the data crosses midnight. If *Midnight* is set to OFF, the date label does not appear when the data crosses midnight. This field is used when the date and time points are graphed on the x-axis.

The default values for XAxis Labeling are defined in the Axis Defaults settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

Page Style = *Font, Style, Size*
The specifications for the text attributes of the text on the page bar of the report window.

Font
Specifies the typeface in which the page bar text is printed.

Style
Specifies the kind of emphasis for the typeface of the page bar text. Some acceptable values are

Normal	For no emphasis
Bold	For highlighting the text
Italic	For emphasizing the text

You can specify multiple styles for the page bar text. However, the styles must be joined by a plus (+) sign. For example, to specify that the page bar text be printed in bold and italic typeface, you enter bold+italic in the Style field of the specification.

Size
Defines the number of points for the typeface of the page bar text. This value determines the height of the characters in which the page bar text is printed. The value must be a positive integer.

Status Style = *Font, Style, Size*
The specifications for the text attributes of the text on the report status bar of the report window.

Font
Specifies the typeface in which the report status bar text is printed.

Style
Specifies the kind of emphasis for the typeface of the report status bar text. Some acceptable values are

Normal	For no emphasis
Bold	For highlighting the text
Italic	For emphasizing the text

You can specify multiple styles for the report status bar text. However, the styles must be joined by a plus (+) sign. For example, to specify that the report status bar text be printed in bold and italic typeface, you enter bold+italic in the Style field of the specification.

Size

Defines the number of points for the typeface of the report status bar text. This value determines the height of the characters in which the report status bar text is printed. The value must be a positive integer.

Point Consolidation = *Enable, Points*

The specifications that determine how the points of the data series are grouped along the horizontal axis. In general, it determines the number of data series points from the database that are consolidated to form one data point on the graph.

Sometimes the number of data points in the data series is more than can actually be graphed in a window. When this happens, TeamQuest View automatically consolidates data points so that the entire time window can be graphed. This concept is called point consolidation.

Enable

Determines if automatic point consolidation is used for the report window. The acceptable values are ON or OFF. ON indicates that automatic point consolidation is in effect. OFF indicates manual consolidation is in effect.

Points

Is an integer greater than zero that specifies the number of data points that are consolidated into one point on the graph when manual consolidation is in effect.

The default values for Point Consolidation are defined in the Miscellaneous settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

Legend = *Enable, Location(, top, left, bottom, right)*

The specifications determining where a legend appears on the report.

Enable

Indicates whether the legend appears on the report. The acceptable values are ON or OFF. If *Enable* is set to ON, the legend appears on the report. If *Enable* is set to OFF, the legend does not appear on the report.

Location

Specifies the location of the legend on the report.

Right	Places the legend to the right of the chart.
Left	Places the legend to the left of the chart.
Top	Places the legend above the chart.
Bottom	Places the legend below the chart.
Float	Places the legend over the top of the chart.

If Location is float, the fields (*top, right, left,* and *bottom*) are required to place the legend properly. If the location is not float, the fields (*top, right, left* and *bottom*) are ignored.

top	Specifies the relative location for the upper coordinate of the top left-hand corner of the window containing the legend.
left	Specifies the relative location for the left coordinate of the top left-hand corner of the window containing the legend.
bottom	Specifies the relative location for the lower coordinate of the bottom right-hand corner of the window containing the legend.
right	Specifies the relative location for the right coordinate of the bottom right-hand corner of the window containing the legend.

Legend Style = *Font, Style, Size, Foreground, Background, Border*

The specifications for the text attributes of the entries of the legend.

Font

Specifies the typeface in which the text is printed.

Style

Specifies the kind of emphasis for the typeface of the text. Some acceptable values are

Normal	For no emphasis
Bold	For highlighting the text
Italic	For emphasizing the text

You can specify multiple styles for the text. However, the styles must be joined by a plus (+) sign. For example, to specify that the text be printed in bold and italic typeface, you enter bold+italic in the Style field of the specification.

Size

Defines the number of points for the typeface of the legend text. This value determines the height of the characters in which the legend text is printed. The value must be a positive integer.

Foreground

Indicates the color in which the text of the legend is printed.

Background
Indicates the color of the area that appears directly behind the legend. The background area is a boxed area and acts as a frame for the legend. If a background color is specified, the boxed area behind the legend covers any graph lines and other text that may appear behind the legend. If the *Background* field for the legend text is not specified, TeamQuest View allows any graph lines or other text that may be behind the legend to show through.

Border
Indicates the color of the border line that appears for the background area of the legend. If the *Border* field is not specified, TeamQuest View does not display a border line around the background area of the legend.

The default values for Legend Style are defined in the Report Styles settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

Threshold Color = *Normal, Exception*
The colors that are used to display thresholds on the report. The colors indicated by these specifications appear in a small box next to the parameter names in the legend of the report.

Normal
Is the color that appears on the report when the data value is above the lower threshold limit and below the upper threshold limit. Valid colors for this field are any of the colors defined in the Threshold settings of the User Preferences.

Exception
Is the color that appears on the report when the data value is below the lower threshold limit or above the upper threshold limit. Valid colors for this field are any of the colors defined in the Threshold settings of the User Preferences.

The default values for Threshold Color are defined in the View Defaults settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

Threshold Pattern = *Normal, Exception*

The patterns that are used to display thresholds on the report. The patterns indicated by these specifications appear in a small box next to the parameter names in the legend of the report.

Normal

Is the pattern that appears on the report when the data value is above the lower threshold limit and below the upper threshold limit.

Exception

Is the pattern that appears on the report when the data value is below the lower threshold limit or above the upper threshold limit.

Valid patterns for Normal and Exception are

Solid	For no pattern
Horizontal	For a horizontal line pattern within the object
Vertical	For a vertical line pattern within the object
ForwardDiagonal	For a line pattern of stripes on a forward diagonal within the object
BackwardDiagonal	For a line pattern of stripes on a backward diagonal within the object
Cross	For a line pattern of square shapes within the object
DiagonalCross	For a line pattern of diamond shapes within the object

The default values for Threshold Pattern are defined in the View Defaults settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

Window Area = *top, left, bottom, right*

The specifications that define the initial area of the report window. The initial window is defined in pixels.

top

Specifies the pixel location for upper coordinate of the top left-hand corner of the window.

left

Specifies the pixel location for the left coordinate of the top left-hand corner of the window.

bottom
Specifies the pixel location for the lower coordinate of the bottom right-hand corner of the window.

right
Specifies the pixel location for the right coordinate of the bottom right-hand corner of the window.

Background Color = *Color*
The color for the background of the report displayed. The default value is obtained from the View Defaults settings of the User Preferences. If a color is not specified, TeamQuest View uses the default value. See the *TeamQuest View User Guide* for information on User Preferences.

BFNote = *Top, Left, Bottom, Right, Vertical, Horizontal, Font, Style, Size, Foreground, Background, Border*
The specifications for the text attributes associated with the best fit equation annotation. The text appears in a rectangle defined by the relative coordinates with a range of 0 through 10,000.

Top
Specifies the relative location for upper coordinate of the top left-hand corner of the best fit annotation box.

Left
Specifies the relative location for the left coordinate of the top left-hand corner of the best fit annotation box.

Bottom
Specifies the relative location for the lower coordinate of the bottom right-hand corner of the best fit annotation box.

Right
Specifies the relative location for the right coordinate of the bottom right-hand corner of the best fit annotation box.

Vertical
Defines the vertical alignment of the text within the best fit annotation box. The acceptable values are

Top	Aligns the text at the top of the best fit annotation box.
Center	Centers the text vertically in the best fit annotation box.
Bottom	Aligns the text at the bottom of the best fit annotation box.

Horizontal
Defines the justification of the text within the best fit annotation box. The acceptable values are

Left	Justifies the text to the left in the best fit annotation box.
Center	Centers the text horizontally in the best fit annotation box.
Right	Justifies the text to the right in the best fit annotation box.

Font
Specifies the typeface in which the best fit annotation text is printed on the report.

Style
Specifies the kind of emphasis for the typeface of the best fit annotation text. Some acceptable values are

Normal	For no emphasis
Bold	For highlighting the text
Italic	For emphasizing the text

You can specify multiple styles for the text. However, the styles must be joined by a plus (+) sign. For example, to specify that the best fit annotation text be printed in bold and italic typeface, you enter bold+italic in the Style field of the specification.

Size
Specifies the height of the characters, measured in points, of the typeface in which the best fit annotation text is printed. The value must be a positive integer.

Foreground
Indicates the color in which the text of the best fit annotation is printed.

Background
Indicates the color of the best fit annotation box that appears directly behind the best fit annotation text. If a background color is specified, the best fit annotation box covers any graph lines and other text that may appear behind this area. If the *Background* field is not specified, TeamQuest View allows any graph lines or other text that may be behind the title to show through.

Border
Indicates the color of the border line that appears for the background area of the best fit annotation text. If the *Border* field is not specified, TeamQuest View does not display a border line around the background area of the best fit annotation text.

The default values for the text attributes are Arial, Normal, 8, and Black.

Note = *Text, Top, Left, Bottom, Right, Vertical, Horizontal, Font, Style, Size, Foreground, Background, Border*

The specifications for annotation that appears on the report. The text of the annotation appears in a rectangle defined by relative coordinates with a range of 0 through 10,000.

Text

Can be up to 80 characters long. If it contains spaces, it must be enclosed in quotation marks (" ").

Top

Specifies the relative location for upper coordinate of the top left-hand corner of the annotation box.

Left

Specifies the relative location for the left coordinate of the top left-hand corner of the annotation box.

Bottom

Specifies the relative location for the lower coordinate of the bottom right-hand corner of the annotation box.

Right

Specifies the relative location for the right coordinate of the bottom right-hand corner of the annotation box.

Vertical

Defines the vertical alignment of the text within the annotation box. The acceptable values are

Top	Aligns the text at the top of the annotation box.
Center	Centers the text vertically in the annotation box.
Bottom	Aligns the text at the bottom of the annotation box.

Horizontal

Defines the justification of the text within the annotation box. The acceptable values are

Left	Justifies the text to the left in the annotation box.
Center	Centers the text horizontally in the annotation box.
Right	Justifies the text to the right in the annotation box.

Font

Specifies the typeface in which the annotation text is printed on the report.

Style
Specifies the kind of emphasis for the typeface of the annotation text. Some acceptable values are

Normal	For no emphasis
Bold	For highlighting the text
Italic	For emphasizing the text

You can specify multiple styles for the text. However, the styles must be joined by a plus (+) sign. For example, to specify that the annotation text be printed in bold and italic typeface, you enter bold+italic in the Style field of the specification.

Size
Specifies the height of the characters, measured in points, of the typeface in which the annotation text is printed. The value must be a positive integer.

Foreground
Indicates the color in which the text of the annotation is printed.

Background
Indicates the color of the annotation box that appears directly behind the annotation text. If a background color is specified, the annotation box covers any graph lines and other text that may appear behind this area. If the *Background* field is not specified, TeamQuest View allows any graph lines or other text that may be behind the title to show through.

Border
Indicates the color of the border line that appears for the background area of the annotation text. If the *Border* field is not specified, TeamQuest View does not display a border line around the background area of the annotation text.

The default values for the text attributes are Arial, Normal, 8, and Black.

Export Data = *Page, Spread, Comma*
The specifications for exporting data from TeamQuest View.

Page
Indicates whether the data from the current page of the report is exported. The acceptable values are ON or OFF. ON indicates that only the data from the current page of the report is exported. OFF indicates that all data from the report is exported.

Spread
Indicates whether the data is exported in spreadsheet format. The acceptable values are ON or OFF. ON indicates that the data is output in spreadsheet format. OFF indicates that the data is output in a database-like format.

Comma

Indicates whether the data is delimited by commas. The acceptable values are ON or OFF. ON indicates that the data is delimited by commas. OFF indicates that the data is delimited by tabs.

Auto Multiplier = {ON | OFF}

Determines if the Auto Multiplier feature is turned on or off. ON indicates the Auto Multiplier feature is turned on. OFF indicates the Auto Multiplier feature is off.

The default value for Auto Multiplier is defined in the View Defaults settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

Display Markers = {ON | OFF}

Determines if display markers appear on the report. ON indicates that the display markers are turned on. OFF indicates that the display markers are turned off.

The default value for Display Markers is defined in the View Defaults settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

Sequence = *Color, Hatch, Marker, Line*

The attributes of the graphic representations are inherited from the multiple sequence statements that appear in the report definition. The order the attributes uses is determined by the order of the sequence statement in the report definition.

Color

Specifies the color that is used for a particular graphic object. All valid colors are defined in Sequences settings of the User Preferences. See the *TeamQuest View User Guide* for information on User Preferences.

Hatch

Specifies the hatch pattern that is used for a particular graphic object. Valid hatch patterns are

Horizontal	For a horizontal line pattern within the object
Vertical	For a vertical line pattern within the object
ForwardDiagonal	For a line pattern of stripes on a forward diagonal within the object
BackwardDiagonal	For a line pattern of stripes on a backward diagonal within the object
Cross	For a line pattern of square shapes within the object
DiagonalCross	For a line pattern of diamond shapes within the object

Marker

Specifies the marker type for a point or line graph. Valid markers are

Invisible	For no marker
Diamond	For a diamond-shaped marker
Triangle	For a triangle-shaped marker
Square	For a square-shaped marker
Circle	For a circle-shaped marker

Line

Specifies the line type on the line graph. The valid line types are solid, Dot, Dash, Medium, or Heavy.

C.5. Parameter-Group Sections

A parameter-group section appears once for each parameter group that is displayed in the report. The parameter-group section explains database information used in the data request to the host. Each parameter group represents an individual data request. A parameter group can create multiple parameter lines.

Format

The format for the parameter-group section is

```
[parameter-group]
Keyname = keyvalue(, keyvalue...)
Value Types = ({Average|StdDev|All}, ...)
Top = Positive-integer, StatisticItem," "{, LastPoint{,[?]}}
Bottom = Positive-integer, StatisticItem," "{, LastPoint,[?]}}
```

The following descriptions explain the syntax of the parameter group specifications of the report definition:

[*parameter-group*]
: The name of the data series. This is a unique name by which the data series is known throughout the report definition. The *parameter-group* can be a string of up to 24 alphanumeric characters.

Keyname = *keyvalue*(, *keyvalue...*)
: The definitions for the host-defined keywords. The host defines the keywords for the *Keyname* specification. If a keyword used for a *Keyname* specification is different from the one defined by the host, the specification is ignored. Some *Keyname* specifications can contain multiple *keyvalue* specifications. If you supply multiple *keyvalues* for a *Keyname* specification that does not allow multiple values, all values but the first value are ignored.

Keyname
: For the TeamQuest Performance Software, the *Keyname* is one of the following keywords defined by the host system:

Statistic-related key names

Category Group
Category
Subcategory
Statistic

Note: *On some platforms, Category and Subcategory are combined into the form Category:Subcategory and called Class and Subclass in the TeamQuest View interface.*

Qualifier-related key names

System
Workload Set
Workload
Resource
Application Group (Unisys OS 2200 only)

keyvalue
Can be a string of up to 52 characters which defines the *Keyname*. The *keyvalue* specifications must be enclosed in quotation marks (" ").

[?]	When this keyword is used within a list of keywords you will be prompted to supply information to complete the field at the time the report is opened. For example, if the [?] appears as the *keyvalue* for the Resource, TeamQuest View displays a list of all legal resources specified for statistic reported. You will not be prompted and the field will be automatically filled in if there is only one choice.
[??]	This keyword is like the [?] keyword, except, it will be always saved in the report definition regardless of the save preferences set in the Miscellaneous settings of User Preferences. This keyword corresponds to the Always Ask button in the prompt dialog boxes.
[??1]	This keyword is like the [?] keyword, except you will be prompted even if there is only one choice. This is used to allow defining other selection settings such as include, exclude, and previous group settings even when there is only one choice in the database you are currently working in.
[?include '*xxx*']	These keywords are used to define an include mask pattern. It is used to include choices from a list based on a mask statement or an exclude mask pattern. The patterns can be made up of any character with the following mask characters available: * Matches zero of more choices. ? Matches exactly one character.
[?exclude '*xxx*']	These keywords are used to define an exclude mask pattern. It is used to exclude choices from a list based on a mask statement or an include mask pattern. The patterns can be made up of any character with the following mask characters available: * Matches zero of more choices. ? Matches exactly one character.

[parameter-group] This keyword allows you to specify the name of a previous *parameter-group* to use the keyword values defined for that group. The referenced *parameter-group* must be a previously defined *parameter-group*.

Value Types = ({Average | StdDev | All}, ...)

The specification for the type of data that is included in the report. Each data sample in the database contains the average and standard deviation values for the specified time interval. You must enter one of the following choices in this field:

Average Specifies that the average value for the data sample be reported.

StdDev Specifies that the standard deviation value for the data sample be reported.

All Specifies that all values for the data sample be reported.

The default value for Value Types is Average.

Top = *Positive-integer, StatisticItem,*" "{, *LastPoint*{,[?]}}

The specifications for selecting the most active resources for a given statistic.

Positive-integer

Specifies the number of the most active resources that are displayed. A [?] can be specified instead of an integer value to allow you to select this value when the report is opened.

StatisticItem

Is a keyword chosen from the Statistic keyword list. A [?] can be specified instead of an integer value to allow you to select this value when the report is opened.

" "

Is for backward compatibility with previous release levels. Leave this setting as a blank string or it may be set to the reserved keyword [?] to cause the interface to prompt you to modify Top *N* settings.

LastPoint

If set to On, only the last point is used to determine the Top *N* order. If set to Off or not set, the entire report calendar range is used to determine the Top *N* order.

[?]

Include the reserved keyword [?] to cause the interface to prompt you to modify Top *N* settings.

Bottom = *Positive-integer, StatisticItem,*" "{, *LastPoint*,[?]}}

The specifications for selecting the least active resources for a given statistic.

Positive-integer

Specifies the number of the least active resources that are displayed. A [?] can be specified instead of an integer value to allow you to select this value when the report is opened.

StatisticItem

Is a keyword chosen from the Statistic keyword list. A [?] can be specified instead of an integer value to allow you to select this value when the report is opened.

" "

Is for backward compatibility with previous release levels. Leave this setting as a blank string or it may be set to the reserved keyword [?] to cause the interface to prompt you to modify Bottom *N* settings.

LastPoint

If set to On, only the last point is used to determine the Bottom *N* order. If set to Off or not set, the entire report calendar range is used to determine the Bottom *N* order.

[?]

Include the reserved keyword [?] to cause the interface to prompt you to modify Bottom *N* settings.

C.6. Individual Parameter Setting Sections

The following subsection defines threshold and multiplier settings on specific parameters or general settings that are applied to parameters with common key values. These settings are applied to the parameters which result from the parameter groups defined in the report definition files.

Note: *These settings are overwritten when the report is saved.*

The format for an individual parameter setting section is

```
[Parm:id]
Keyname = keyvalue
Value Types = {Average|StdDev}
Threshold = EnableLow, Lower(, EnableUp, Upper)
Parameter = {On|Off}, Display, Multiplier
Lower Threshold Process = "executable name"
Upper Threshold Process = "executable name"
```

The following descriptions explain the syntax of the individual parameter setting specifications of the report definition:

[Parm: *id*]
This is a unique name for an individual parameter setting. The individual parameter setting can be a string of up to 24 characters. The name must begin with Parm: and *id* is a numerical value, which makes it unique from other individual parameter groups.

Keyname = keyvalue
The definitions for the host-defined keywords, which identify the parameter to which the settings are applied. All *Keynames* must be specified to match a related parameter.

Keyname
For the TeamQuest Performance Software, the *Keyname* is one of the following keywords defined by the host system:

Statistic-related key names

Category Group
Category
Subcategory
Statistic

Note: *On some platforms, Category and Subcategory are combined into the form Category:Subcategory and called Class and Subclass in the TeamQuest View interface.*

Qualifier-related key names

System
Workload Set
Workload
Resource
Application Group (Unisys OS 2200 only)

keyvalue
Can be a string of up to 52 characters which defines the *Keyname*. The *keyvalue* specifications must be enclosed in quotation marks (" ").

[?] The *keyvalue* of [?] signifies that this *Keyname* is not considered when matching the *Keyname* values with the actual parameter values to set the threshold and parameter settings. For example, if the [?] appears as the *keyvalue* for the Resource, then the threshold and parameter attributes will be applied to all parameters that match all the other *Keyname* values, ignoring the Resource *Keyname*.

Value Types = {Average | StdDev}

The specification for the type of data that is included in the report. This must match the Value Types setting of the parameters you want to match. The Value Types can be one of the following:

Average

Specifies that the average value for the data sample be reported.

StdDev

Specifies that the standard deviation value for the data sample be reported.

Threshold = *EnableLow*, *Lower*(, *EnableUp*, *Upper*)

The threshold limits for the parameter value of this parameter. When parameter values exceed the specified limits, the color or pattern displayed in the threshold box in the report legend changes according to the specified values in the Threshold Color and Threshold Pattern statements of the general section.

EnableLow

Indicates whether the lower threshold limit is activated for this parameter. If *EnableLow* is set to On, the threshold box appears on the report legend. If *EnableLow* is set to Off, the lower threshold is not active for this data series. If both *EnableLow* and *EnableUp* are set to Off, the threshold box does not appear on the report legend. The default for this field is Off.

Lower

Specifies the lower threshold limit. When parameter values are less than or equal to this value, the threshold box for the data series appears in the color or pattern specified in the *Normal* field of Threshold Color or Threshold Pattern as defined in the general section.

EnableUp

Indicates whether the upper threshold limit is activated for this parameter. If *EnableUp* is set to On, the upper threshold appears on the report legend. If *EnableUp* is set to Off, the upper threshold is not active for this data series. If both *EnableLow* and *EnableUp* are set to Off, the threshold box does not appear on the report legend. The default for this field is Off.

Upper

Specifies the upper threshold limit. When parameter values are greater than or equal to this value, the threshold box for the data series appears in the color or pattern specified in the *Exception* field of Threshold Color or Threshold Pattern as defined in the general section.

Parameter = [On | Off], *Multiplier*

Identifies if this parameter is currently displayed and what multiplication factor is applied to all values of this parameter. The acceptable values are On or Off. On indicates that the parameter is displayed. Off indicates that the parameter is hidden.

Multiplier

Specifies the multiplication factor applied to all values of the parameter.

Lower Threshold Process = "*executable name*"

The alarm program that is specified for the lower threshold limits for the parameter values of this parameter. When parameter values exceed the specified limits, the alarm program is invoked according to the values specified in the Sequence statement of the general section.

Upper Threshold Process = "*executable name*"

The alarm program that is specified for the upper threshold limits for the parameter values of this parameter. When parameter values exceed the specified limits, the alarm program is invoked according to the values specified in the Sequence statements of the general section.

Appendix D
Host-Based Reporting for Aggregation Set Data

TeamQuest Baseline provides the capability to generate reports from the host system without using the TeamQuest View interface. The **tqrpt** program can generate a report based on a report definition stored in a file or can interactively solicit the information required to generate a report. The printed reports are normally 132 characters wide, with a separate column for each parameter reported.

Note: *The **tqrpt** program is not available for TeamQuest Online.*

The processor call has the following format:

```
@tqrpt [-d,databasename,] [-a,aggset,];
       [-s,starttime,] [-e,endtime,] [-c,time-consolidation,];
       [-f,firstdate,] [-l,lastdate,] [-p,];
       [-w,pagewidth,] [-x,delimiter,] [-r,report,] [-o,filename,];
       [-v]
```

where:

-d specifies the database from which you want to generate a report. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the Database setting in the tq-ini element is used (see 3.1).

-a specifies the aggregation set to use for retrieving data. The *aggset* parameter is the name of an existing aggregation set. If no aggregation set is specified, **tqrpt** uses the aggregation set with the finest granularity.

-s specifies the start time for the data. The *starttime* parameter is in the form of *hhmmss*. The default is 000000.

-e specifies the end time for the data. The *endtime* parameter is in the form of *hhmmss*. The default is 235959.

-c specifies the number of minutes to consolidate for each point. The *time-consolidation* parameter must be an integer. This number will be rounded down to the nearest whole interval stored in the aggregation set.

-f specifies the starting date for the data. The *firstdate* parameter is expressed as *mmddyyyy*, *mmddyy,* or *mmdd*. The current date is the default.

If a 2-digit year is specified, the digits 00 through 27 are interpreted as the years 2000 through 2027 and the digits 28 through 99 are interpreted as the years 1928 through 1999.

You can express the *firstdate* parameter as a date relative to the current date by entering *n-*, where *n* is the number of days before the current date.

-l specifies the ending date for the data. The *lastdate* parameter is expressed as *mmddyyyy*, *mmddyy,* or *mmdd*. The current date is the default.

If a 2-digit year is specified, the digits 00 through 27 are interpreted as the years 2000 through 2027 and the digits 28 through 99 are interpreted as the years 1928 through 1999.

You can express the *lastdate* parameter as a date relative to the current date by entering *n-*, where *n* is the number of days before the current date.

-p prints a list of stored and derived parameters and then exits the program.

-w specifies the number of characters to print per line. The default value is 132.

-x specifies that the report output is to be in a condensed format that is suitable for import into a spreadsheet program. The *delimiter* parameter is c for comma-delimited output or t for tab-delimited output.

-r specifies the name of a file containing the report definition. If you specify the -r option, it must be the last parameter on the processor call.

-o specifies the name of the file that receives the generated report.

-v specifies that the output values in the report should be displayed with 5 decimal places. This option only effects reports generated using the -x option (condensed format). If the -v option is not used, the values in the report will be displayed with 2 decimal places.

All parameters are optional. If a report definition file is specified, the -a, -s, -e, -c, -f, and -l options override the values specified in the report definition file. If no report definition file is specified, the option values are used and that item is not prompted.

The report definitions have a fixed format. The information presented in them is positionally dependent and must be in the following form:

```
aggregation_set_name
report title
start_time
end_time
time_consolidation
first_date
last_date
date_consolidation
parameter_selection1
.
.
.
parameter_selectionN
```

The date_consolidation specifies the number of dates to consolidate for each point.

The parameter_selection definition can be repeated as many times as needed. The parameter_selection definition is specified using the parameter name all on a single line with the key names separated by colons. The order of the key names is

```
system:category group:category:subcategory:statistic:
resource:application group:workload set:workload
```

The parameter_selection definition is case-sensitive. All colons must be included even if a parameter key name is blank.

A single mask character (*) can be specified in each key name field of the parameter selection. The mask can be specified as "**string*", "*string**", or "*string***string*" where *string* is any string of characters.

To see a list of parameters currently defined in the database, enter a question mark (?) at the parameter selection prompt.

You terminate parameter selection prompting by entering a blank transmit at the parameter selection prompt. If you enter any type of control statement (including @EOF) as a response to a prompt, **tqrpt** terminates immediately without producing a report.

D.1. Using tqrpt for Condensed Output

When the -x option is used with **tqrpt**, the report output is in a condensed format with comma or tab delimiters. This format is suitable for importing into a spreadsheet program.

The length of a line in the condensed output is controlled by two factors, a maximum field count and a maximum line width in characters. In comma-delimited output, each of the fields are separated by commas. In tab-delimited output, the fields are separated by tab characters. A maximum of 255 fields are allowed on each output line. Since date and time take up two fields, you can have up to 253 parameters in a report using the -x option.

The -w option of **tqrpt** also controls the maximum width of an output line. For condensed output, the default line width is 4,000 characters. By using the -w option, the line width can be decreased or it can be increased to a maximum of 8,000 characters. When either the field or character maximum is reached on a line, any remaining output for that line is discarded. Truncation of a line will not result in partial field values. For example, if the last field of a line will not entirely fit within the line limits, the complete field is truncated for all lines on the report.

The condensed report output consists of a set of a parameter definition lines followed by a series of detail lines. The parameter definition lines contain the names of the parameters selected for the report. The parameters are listed in the order that they were selected with each parameter key name on a separate line. The detail lines contain the data values for each of the parameters for the selected time period. Each detail line begins with date and time fields that identify the data values on that line.

For example, the following parameters were selected for reporting from the 30-minute aggregation set on May 8, from 12:00 to 14:00:

```
system:category group:category:subcategory:statistic:resource:
     appgrp:workload set:workload
sysa:Processor:Total:Utilization:Batch::::
sysa:Processor:Total:Utilization:Demand::::
sysa:IO Activity:Disk:Summary:Percent Busy:DSKF1:::
sysa:System Log:Workload:Batch:IO SUPs:::primary:OTHER
```

The report would appear as follows:

```
System,,sysa,sysa,sysa,sysa
Category Group,,Processor,Processor,IO Activity,System Log
Category,,Total,Total,Disk,Workload
Subcategory,,Utilization,Utilization,Summary,Batch
Statistic,,Batch,Demand,Percent Busy,IO SUPs
Resource,,,,DSKF1,
Appgrp,,,,,
Workload Set,,,,,primary
Workload,,,,,OTHER
Date,Time

01/13/2012,12:00:00,45.00,10.00,20.00,15.00
01/13/2012,12:30:00,40.00,10.00,15.00,10.00
01/13/2012,13:00:00,50.00,10.00,12.00,10.00
01/13/2012,13:30:00,45.00,10.00,20.00,17.00
01/13/2012,14:00:00,40.00,10.00,10.00,5.00
```

D.2. Using tqrpt for Reporting Derived Statistics

You can use **tqrpt** to report the values of derived statistics you have defined in your database. Derived statistic definitions are saved in the database as formulas. The actual data values reported for a derived statistic are not stored in the database, but are calculated at report time.

You use the p option of **tqrpt** to generate a list of the stored parameters and derived statistics that exist in your database. The derived statistic names are made up of the Category Group, Category, Subcategory, and Statistic keys. To report on derived statistics, you enter these four keys as part of a parameter_selection definition just as you do when reporting stored parameters. In addition, you must enter a system name and you may need to include the resource, workload set, workload, and application group names on the parameter_selection definition for the derived statistic.

When you define a derived statistic formula, aliases are used to refer to other parameters in the database. Substitution indicators are used in aliases to indicate that certain key values are to be specified at report time. For example, you may use a substitution indicator for the resource name key of a given alias. To report on the derived statistic, you would specify the resource as part of the parameter_selection definition entered at the **tqrpt** prompt. For more information on derived statistic referencing, aliases, and substitution indicators, see 2.6.

Example

To illustrate the use of aliases, perhaps you have defined a derived statistic in your database as follows:

Derived statistic name	Derived:Derived Category:Derived Subcategory:processory-busy
Aliases	idle = $SYSTEM:Processor:Processor:Utilization:Idle:$RES:::
Formula	100 - idle

To report on the derived statistic, processor-busy, you first need to determine the system and resource (that is, IP) names in which you are interested. To report processor-busy on IP1 and IP2 for system venus, you would enter the following two lines at the **tqrpt** parameter_selection prompt:

```
venus:Derived:Derived Category:Derived Subcategory:processor-busy:IP1:::
venus:Derived:Derived Category:Derived Subcategory:processor-busy:IP2:::
```

When each of the derived statistics is calculated, venus is substituted as the system name in the alias. IP1 and IP2 are substituted as the resource name in the alias.

Rules for Parameter_Selection Definition

The rules for the parameter_selection definition are the same as with stored parameters. All keys must be specified in order. Keys are case-sensitive. All colons must be included even if a key name is blank. You must be sure to specify the key name for substitution in any derived statistic formula aliases. As with stored parameters, you can use the mask character (*) to specify any of the keys. For example, the following parameter_selection would generate a report of processor-busy for each IP on your system:

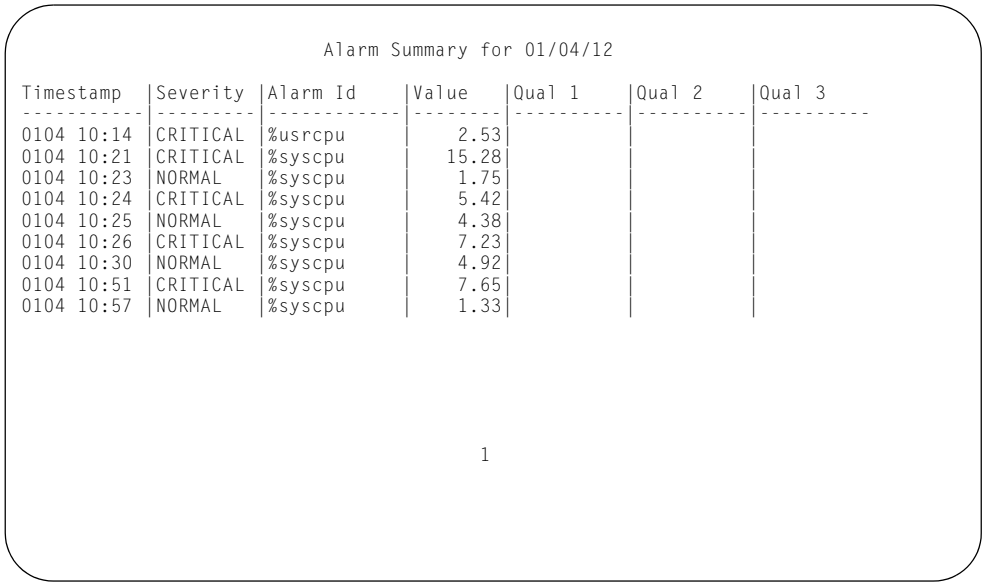
```
venus:Derived:Derived Category:Derived Subcategory:processor-busy:*:::
```

Appendix E
Host-Based Reporting for Table Data

E.1. Overview of tqrep

TeamQuest Baseline provides the capability to generate reports using the table records stored in your performance database. The **tqrep** program can generate a report based on a user-defined report definition stored in a file or by using one of the default reports definitions provided for the table. The **tqrep** program is run on the host system and allows you to generate reports without using the TeamQuest View user interface.

Figure E–1 shows an example of the type of report that can be generated using **tqrep**.



Alarm Summary for 01/04/12

Timestamp	Severity	Alarm Id	Value	Qual 1	Qual 2	Qual 3
0104 10:14	CRITICAL	%usrcpu	2.53			
0104 10:21	CRITICAL	%syscpu	15.28			
0104 10:23	NORMAL	%syscpu	1.75			
0104 10:24	CRITICAL	%syscpu	5.42			
0104 10:25	NORMAL	%syscpu	4.38			
0104 10:26	CRITICAL	%syscpu	7.23			
0104 10:30	NORMAL	%syscpu	4.92			
0104 10:51	CRITICAL	%syscpu	7.65			
0104 10:57	NORMAL	%syscpu	1.33			

1

Figure E–1. Report Generated by tqrep

E.2. Command Format

The **tqrep** command line has the following format:

```
@tqrep [-h,] [-d,databasename,] [-l,];
     [-s,starttime,] [-e,endtime,] [-c,time-consolidation,];
     [-t,tablename,] [-r,] [-z,] [-n,defRepname,];
     [-f,reportdefnfile,] [-o,outputfile,] [-x,delimiter,];
     [-w,reportwidth,];
     END
```

where:

-h prints usage information containing the command line syntax.

-d specifies the database from which you want to generate a report. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the default database is used.

-l provides information about the available tables in the database. Included in the information for each table is a list of the default report names for the table and a list of the field names and type of data contained in the field.

-s specifies the start time for the data. The *starttime* parameter is in the form of *MMDDYYYY:HHMMSS*. If no value is specified, the StartTime value from the report definition is used.

-e specifies the end time for the data. The *endtime* parameter is in the form of *MMDDYYYY:HHMMSS*. If no value is specified, the EndTime value from the report definition is used.

-c specifies the time consolidation for the selected records. The *time-consolidation* parameter has the format of *nn*HOUR, *nn*MIN, or *nn*SEC where *nn* is the number of units. If no value is specified, the TimeCons value from the report definition is used.

-t specifies the name of the table to report on. The *tablename* parameter is expressed as *class:name*.

For example, in the table name "Alarm:Alarm Log", Alarm is the *class* and Alarm Log is the *name*. If the table name contains a space as in the example Alarm Log, it must be enclosed in double quotation marks (" "). If the table name does not contain any spaces, the quotation marks are not needed.

-r prints the default report for the table specified by the -t option. If you use the -r option, you must specify the -t option.

-z prints the report definition of the first default report for the table specified by the -t option. If you use the -z option, you must specify the -t option. The -z option can be used to create an initial report definition suitable for editing.

-n prints the specified report for the table specified by the -t option. The *defRepname* parameter is the name of a report that has been defined for the table. If you use the -n option, you must specify the -t option.

To find a list of the available report names for a table, use the -l option.

-f prints the report from the specified report definition file. The *reportdefnfile* parameter specifies the name of a file that contains a report definition.

-o prints the report to the specified output file. If this option is not specified, the output file specified in the report definition is used.

-x exports the data using the specified delimiter. The *delimiter* parameter can be any printable character string. In addition, the reserved words TAB, COMMA, and SPACE can be used to indicate a tab-, comma-, or space-delimited report, respectively.

-w specifies the number of characters to use for the report width. Any text outside the specified width is not printed.

Note: *You must request some type of reporting for each* ***tqrep*** *execution. The -t, -l, or -f option must be specified to indicate the type of reporting to be performed.*

Example

The following command was used to generate the report in Example E–1. The -f option specifies the report definition file alarm.rep.

```
@tqrep -f,alarm-rep., end
```

E.3. Report Definition File Format

A report definition file is used to define the content and layout of the report generated by **tqrep**. A report definition contains two or more sections: a single Report section and one or more Table sections. A section consists of the section name followed by section keys with assigned values.

The following conventions apply to a report definition file:

- The section names are enclosed in brackets ([ ]).
- A section continues until the next section or the end of the report file is reached.
- The lines in a report definition can be continued to the next by using a backslash (\) as the last character in the line.
- If strings in the report definition must be enclosed in double quotation marks (" "), a double quotation mark can be placed in the quoted string by placing a backslash (\) in front of the embedded quotation mark. For example:

```
filter = "login!= \"root\""
```

The following example shows the report definition file alarm-rep, which was used to generate the report in Example E–1.

```
[Report]                                                                    A
OutputFile = "alarm-txt."                                                   B
StartTime = &NDATE:000000
EndTime = &NTDATE
PageLength = 20
PageWidth = 80
CenterHeader = "Alarm Summary for &BDATE"
CenterFooter = "&PAGE"
[Table]                                                                     C
MaxRecs = 500                                                               D
DataFields = Timestamp, "MMDD hh:mm ",, "Timestamp"
DataFields = Severity, "LJ9",       ,   "Severity"
DataFields = Alarm_ID, "LJ12",      ,     Alarm Id"
DataFields = Parameter_Value, "######.##",  ,  Value"
DataFields = Qualififer_1, "LJ10",    ,    Qual 1"
DataFields = Qualififer_2, "LJ10",    ,    Qual 2"
DataFields = Qualififer_3, "LJ10",    ,    Qual 3"
Sort = Timestamp,Ascending
```

The following lettered descriptions correspond to the capital letters in the preceding example. The letters are not part of the actual screen display.

A This is the first line of the report definition file and the beginning of the Report section. The section name must be enclosed in brackets ([ ]) and is not case-sensitive.

Only one Report section can appear in a report definition file.

The Report section contains information that is applied to all the tables in the report. The Report section defines such things as the name for the output file, the time range of the data, the time consolidation, the page size of the report, the report headings and footers, and the report column delimiters.

B The Report section name is followed by section keys with their assigned values. This line begins the Report section keys. See E.3.1 for a description of the valid keys for the Report section.

C This line begins a Table section. The section name must be enclosed in brackets ([ ]) and is not case-sensitive.

A report definition can contain one or more table sections. Each table section specifies the name of the table and the fields from the table that should be reported on.

D The Table section name is followed by section keys with their assigned values. This line begins the Table section keys. See E.3.2 for a description of the valid keys for the Table section.

E.3.1. Report Section Keys

The keys listed in this section are valid in the Report section of a report definition file. The key names are not case-sensitive. Unless otherwise noted, the key names can be specified only once. If a key is specified more than once, the last key value specified is used.

Database="*database*"
: The name of the database to use. The *database* must be enclosed in double quotation marks (" ").

  This specification can be overridden by the -d option on the **tqrep** command line. If the Database key is not specified in the report definition file and the -d option is not given on the command line, the default database is used.

OutputFile= "*filename*"
: The name of the file where the report is to be written. The *filename* must be enclosed in double quotation marks (" ").

  This specification can be overridden by the -o option on the **tqrep** command line. If OutputFile key is not specified in the report definition file and the -o option is not given on the command line, the report is written to the standard output file.

StartTime=*MMDDYYYY*[:*HHMMSS*]
: The starting date and time for the report.

  This specification can be overridden by the -s option on the **tqrep** command line. If the time *HHMMSS* is not specified, 000000 is assumed.

  The special identifier &NDATE can be used to specify the current date. &NTIME can be used to specify the current time. &NTDATE can be used to specify the current date and time.

EndTime=*MMDDYYYY*[:*HHMMSS*]
: The ending date and time for the report.

  This specification can be overridden by the -e option on the **tqrep** command line. If the time *HHMMSS* is not specified, 000000 is assumed.

  The special identifier &NDATE can be used to specify the current date. &NTIME can be used to specify the current time. &NTDATE can be used to specify the current date and time.

TimeCons=*nn units*
: The time consolidation to use on reported data. *nn* is the number of units. *units* can be HOUR, MIN, or SEC.

  This specification can be overridden by the -c option on the **tqrep** command line. If TimeCons key is not specified in the report definition file and the -c option is not given on the command line, no time consolidation is performed.

PageLength=*nn*
: The length of the printed page in lines. *nn* is the number of lines. If the PageLength key is not specified, 60 lines is assumed.

PageWidth=*nn*
: The width of the printed page in character positions. *nn* is the number of character positions. The page width can be overridden by the -w option on the **tqrep** command line. If the PageWidth key is not specified in the report definition file and the -w option is not given on the command line, 80 characters is assumed.

LeftHeader="*text*"
: A left-justified heading for the report. The *text* must be enclosed in double quotation marks (" ").

  Any of the special identifiers can be used in the heading text.

  The report heading appears at the top of the page and is repeated on every page of the report.

CenterHeader="*text*"
: A centered heading for the report. The *text* must be enclosed in double quotation marks (" ").

  Any of the special identifiers can be used in the heading text.

  The report heading appears at the top of the page and is repeated on every page of the report.

RightHeader="*text*"
: A right-justified heading for the report. The *text* must be enclosed in double quotation marks (" ").

  Any of the special identifiers can be used in the heading text.

  The report heading appears at the top of the page and is repeated on every page of the report.

LeftFooter="*text*"
: A left-justified footer for the report. The *text* must be enclosed in double quotation marks (" ").

  A footer appears at the bottom of the page and is repeated on every page of the report.

CenterFooter="*text*"
: A centered footer for the report. The *text* string must be enclosed in double quotation marks (" ").

  A footer appears at the bottom of the page and is repeated on every page of the report.

RightFooter="*text*"
: A right-justified footer for the report. The *text* must be enclosed in double quotation marks (" ").

  A footer appears at the bottom of the page and is repeated on every page of the report.

ReportTitle=(*row,col*),"*text*"
: A title for the report that is printed on the first page of the report at the specified row and column.

  The (*row,col*) indicates the row and column on the page where the text string should start. The parentheses must be included in the (*row,col*) specification. You can specify the row or column number in one of two ways. To indicate the absolute row or column, specify only a number for the row and column. To indicate a relative position, the number should be preceded by a plus sign (+). The relative position is calculated from the position where the previous character was printed on the page.

  Optionally, you can indicate just a row specification by (*row*). In this case, the *text* is centered on the page.

  The *text* must be enclosed in double quotation marks (" ").

  You can specify more than one ReportTitle key.

ColumnHeadings={YES|NO}
: Determines if column headings appear on all pages of the report. If set to YES, the column headings appear on all pages of the report. If set to NO, the column headings are printed on the first page of the report only. The report heading and footer continue to be printed on each page.

Delimiter={COMMA|SPACE|TAB|"*text*"}
: The delimiter to be used between columns in the report. The delimiter and be one of the reserved words TAB, SPACE, or COMMA, or it can be any text string.

  If a *text* string is used, it must be enclosed in double quotation marks (" ").

Export={COMMA|SPACE|TAB|"*text*"}
: Prints only the data. All other header, footer, and title information if left off the resulting report. In addition, the column headings are printed only once and all page sizes are ignored. Use this key for exporting data that is intended to be imported into a spreadsheet or a database package.

  The Export key specifies the delimiter to be used between fields in the report. The delimiter can be one of the reserved words TAB, SPACE, or COMMA, or it can be any text string.

  If a *text* string is used, it must be enclosed in double quotation marks (" ").

  The Export key can be overridden by the -x option on the **tqrep** command line.

E.3.2. Table Section Keys

The keys listed in this section are valid in the Table section of a report definition file. The key names are not case-sensitive. Unless otherwise noted, the key names can be specified only once. If a key is specified more than once, the last key value specified is used.

TableName=*class*:*name*
: The name of the table to be reported on. This key is required in a table section.

  For example, in the table name "Alarm:Alarm Log", Alarm is the *class* and Alarm Log is the *name*. If the table name contains a space as in the example Alarm Log, it must be enclosed in double quotation marks (" "). If the table name does not contain any spaces, the quotation marks are not needed.

TableTitle=[(*row,col*),]"*text*"
: A title that appears on the first page where data from this table is printed. The table title does not appear on subsequent pages of the same table. The *text* must be enclosed in double quotation marks (" ").

  The (*row,col*) is optional and indicates the row and column on the page where the text string should start. The parentheses must be included in the (*row,col*) specification. You can specify the row or column number in one of two ways. To indicate the absolute row or column, specify only a number for the row and column. To indicate a relative position, the number should be preceded by a plus sign (+). The relative position is calculated from the position where the previous character was printed on the page.

  Optionally, you can indicate just a row specification by (*row*). In this case, the *text* is centered on the page.

  You can specify more than one TableTitle key.

TextFields=[(*row,col*),]"*text*"
: A text string containing informational text that is displayed at the beginning of the table section of the report. The *text* must be enclosed in double quotation marks (" ").

  The (*row,col*) is optional and indicates the row and column on the page where the text string should start. The parentheses must be included in the (*row,col*) specification. You can specify the row or column number in one of two ways. To indicate the absolute row or column, specify only a number for the row and column. To indicate a relative position, the number should be preceded by a plus sign (+). The relative position is calculated from the position where the previous character was printed on the page.

  Optionally, you can indicate just a row specification by (*row*). In this case, the *text* is centered on the page.

  You can specify more than one TextField.

DataFields=*fieldname*,["*format*"],[(*row,col*)],["*text*"]
Specifies how the data from the table is displayed in the report.

The *fieldname* must be a valid field name for the table.

The *format* specifies how the data for the specified field is displayed. The *format* is optional and must be enclosed in double quotation marks (" "). It can be one of the following:

LJ*n*	The data is left-justified and *n* characters are displayed.
RJ*n*	The data is right-justified and *n* characters are displayed.
YY	The year is displayed as a 2-digit number.
YYYY	The year is displayed as a 4-digit number.
MM	The month is displayed as a 2-digit number.
MMM	The month is displayed as a 3-character string.
DD	The day of the month is displayed as a 2-digit number.
hh	The hour is displayed as a 2-digit number.
mm	The minutes are displayed as a 2-digit number.
ss	The seconds are displayed as a 2-digit number.
##	The data is displayed as a decimal number where each number sign (#) represents one digit.
#.#	The data is displayed as a floating-point number where each number sign (#) represents one digit.

The *text* is the heading for the column of values. The *text* must be enclosed in double quotation marks (" "). If the text specification is not included, the column heading defined for the table is used.

The (*row,col*) is optional and indicates the row and column on the page where the text string should start. The parentheses must be included in the (*row,col*) specification. You can specify the row or column number in one of two ways. To indicate the absolute row or column, specify only a number for the row and column. To indicate a relative position, the number should be preceded by a plus sign (+). The relative position is calculated from the position where the previous character was printed on the page.

Optionally, you can indicate just a row specification by (*row*). In this case, the *text* is centered on the page.

If the position of the column is not specified, the column is placed next to the previous column from the table. If this is the first column, the column starts in the left-most position on the page.

You can specify more than one DataFields key.

MaxRecs=*nnn*
The maximum number of records to print. Any records after this number will be discarded.

Sort=*field1*{,ASCENDING|DESCENDING}
[*field2*{,ASCENDING|DESCENDING}] ... \
[*fieldn*{,ASCENDING|DESCENDING}] \

The sort criteria for the records. *field1*, *field2*, ... and *fieldn* are valid field names for the table.

The data will be sorted by the specified field in either ascending or descending order. If the sorted order results in duplicate sort keys, the second field, if specified, will be used to sort these records, and so on.

Filter="*text*"
The criteria to be used to filter records for display. Only records that match the criteria defined in the "*text*" filter statement. The filter statement must be enclosed in double quotation marks (" "). See 5.2.2 for a description of the definition language that can be used to create filter statements.

Combine=*field1*[,*field2*, ...]
The fields to use for combining data records. *field1*, *field2*, ... can be any valid field name for the table. The records will be combined where the values of the given field name match exactly.

E.3.3. Using Special Identifiers in Section Keys

Special identifiers can be used as key values in both the Report section and the Table section of the report definition file. The special identifiers allow for substitution of values at the time the report is generated. By using special identifiers, you do not need to supply actual values in the report definition.

A special identifier consists of an ampersand (&) followed by a valid identifier name. If an ampersand is followed by an unknown identifier, the ampersand and identifier are printed in the report as they appear in the report definition. Special identifiers are not case-sensitive.

The following are valid special identifiers:

&BTIME
The beginning time of the report. The format of the time is the locale default format for time.

&BDATE
The beginning date for the report. The format of the date is the locale default format for date.

&BTDATE
The beginning time and date for the report. The format of the time and date is the locale default format for time and date.

&ETIME
The ending time for the report. The format of the time is the locale default format for time.

&EDATE
The ending date for the report. The format of the date is the locale default format for date.

&ETDATE
The ending time and date for the report. The format of the time and date is the locale default format for time and date.

&NTIME
The current (now) time. The format is *HHMMSS*, which is suitable for report time range specification.

&NDATE
The current (now) date. The format is *YYYYMMDD*, which is suitable for the report date range specification.

&NTDATE
The current (now) time and date. The format is YYYYMMDD:HHMMSS, which is suitable for the report time range specification.

&PAGE
The current page being generated.

&USER
The login name of the user generating the report.

&SYSTEM
The name of the system where the report is being generated.

&TABLE
The name of the table for which data is being printed.

E.4. Using Default Reports

Each table in your performance database can have one or more default reports defined for it. To see a list of the default reports available for a table, use the -l option of **tqrep**. You can use the default reports in the following ways:

- Generate your report using one of the default reports. The -n option of **tqrep** allows you to specify the name of the report you want to generate.
- Place a copy of one of the default report definitions in a file by using the - z option. You can edit this file to create a new report definition. Then use the new report definition file as input to **tqrep** by using the -f option.

E.5. Exporting Data

The **tqrep** program allows you to export data rather than printing a formatted report. The resulting output file can be used as input to a spreadsheet program. To export data, enter the following command:

```
@tqrep -f,alarm-rep,-x,comma, end
```

The resulting output file would look like this:

```
Timestamp,Severity.Alarm Id,Value,Qual 1,Qual 2,Qual 3
0104 10:14,CRITICAL,%usrcpu,2.53,,,
0104 10:21,CRITICAL,%syscpu,15.28,,,,
0104 10:23,NORMAL,%syscpu,1.75,,,,
0104 10:24,CRITICAL,%syscpu,5.42,,,,
0104 10:25,NORMAL,%syscpu,4,38,,,,
0104 10:26,CRITICAL,%syscpu,7.23,,,,
0104 10:30,NORMAL,%syscpu,4.92,,,,
0104 10:51,CRITICAL,%syscpu,7.65,,,,
0104 1057,NORMAL,%syscpu,1.33,,,,
```

Appendix F
The tqgetm Program for TeamQuest Model

The **tqgetm** program allows you to extract statistical data from the performance database in the format required by TeamQuest Model. With the **tqgetm** program, you select the database, workload set, system, and the time and date period that you want to model. **tqgetm** retrieves the required data from the performance database and writes the data to a file that can be used as input to TeamQuest Model. See the *TeamQuest Model User Guide* for information on how to use the **tqgetm** program.

Note: *The **tqgetm** program requires a special access/license key to operate. Contact TeamQuest Corporation to receive a key.*

Appendix G
Alarm History Table

Alarms raised due to threshold violations are handled by the **tqalm** program. The **tqalm** program writes each alarm to the alarm history table. The **tqzip** program can be used to export alarm messages that were generated within a specified time interval.

The alarm history table is automatically created by the **tqalm** program. The alarm history table is created in file TQDATA$**xxxx*alarmlog. The *xxxx* is the first four characters of the database name for which the alarm history table is being created. The following are details of the alarm history table:

Table Field Hierarchy

Class:	Alarm
Name:	Alarm Log
File name:	samfile
Collection interval:	00:00:00–23:59:59
Default retention:	1 month

Data Item	Type	Description
Timestamp	Integer	The date and time at which the alarm occurred
Sequence_Number	Integer	A sequential number assigned to the alarm for uniqueness
System	String	The name of the system for which the alarm occurred
Severity	String	The severity of the alarm: NORMAL, WARNING, or CRITICAL
Alarm_ID	String	The user-defined alarm identifier assigned to this alarm
Qualifier_1	String	Either the resource name (for example, diskname) or application group name for which the alarm was generated. If both resource name and application group name apply to the statistic, Qualifier_1 contains both names separated by a space.
Qualifier_2	String	The workload set name for which the alarm was generated
Qualifier_3	String	The workload name for which the alarm was generated

Statistic	String	The name of the statistic, including the category group, category, subcategory, and statistic name
Parameter_Value	Real Number	The parameter value that caused the alarm to be generated
User_Text_1	String	The user text line 1 from the threshold definition for the alarm
User_Text_2	String	The user text line 2 from the threshold definition for the alarm
User_Text_3	String	The user text line 3 from the threshold definition for the alarm
First_Threshold	Real Number	The first threshold from the threshold definition for the alarm
Second_Threshold	Real Number	The second threshold from the threshold definition for the alarm
Count	Integer	A count of alarms that is used for consolidating the alarms at report time
Parameter_Name	String	The name of the parameter for which the alarm was generated. This includes system, category group, category, subcategory, statistic, resource, application group, workload set, and workload.
Database_Name	String	The name of the performance database that contains the data that was used to detect this alarm condition

Appendix H
tqzip Processor Call Format

The **tqzip** program has the following processor call format:

```
@tqzip [-h,] [-d ,databasename,] [-a ,action,];
       [-o ,parm-file] [-p ,parm-file];
       [-l,] [-u,] [-e,] [-c ,tablespec-file,];
       [-r ,reduction-file,] [-s ,derived-file,];
       [-t ,threshold-file,] [-w ,workload-file]
```

where:

-h prints usage information.

-d specifies the database that **tqzip** should use. The *databasename* parameter is the name of a database that has been created on the host system. If the -d option is not specified, the default database is used.

-a indicates the action to be taken with the threshold checking setting. *action* can be one of the following:

ON turns threshold checking on

OFF turns threshold checking off

LIST displays the current threshold checking setting

See 7.15 for more information on turning on and off threshold checking.

-o outputs the database parameter list to the file specified by the *parm-file* parameter. See 10.8.

-p deletes parameters from the database using the file specified by the *parm-file* parameter. See 10.9.

The *parm-file* used with the -p option must be in the same format as the file produced by the -o option. All key values must be specified in each parameter reference.

-l specifies that a load operation is to be performed. This option must be used with one of the following options: -c, -r, -s, -t, or -w. Information contained in the file specified by the -c, -r, -s, -t, or -w options is loaded into the database.

-u specifies that an unload operation is to be performed. This option must be used with one of the following options: -c, -r, -s, -t, or -w. Information contained in the file specified by the -c, -r, -s, -t, or -w options is unloaded from the database.

-e is used in combination with the -l option when loading workload, reduction, threshold, or derived statistic definitions. If you specify the -e option, the new definitions are loaded without first deleting the existing definitions in the database. If you do not specify the -e option, all existing definitions are deleted from your database before the new ones are loaded.

-c specifies the file that **tqzip** should use for a table collection/retention load or unload operation. The *tablespec-file* parameter is the name of the file containing the data collection and retention specifications for your database. The -c option must be used with the -l or -u option. See 11.10 and 11.11.

-r specifies the file that **tqzip** should use for a reduction load or unload operation. The *reduction-file* parameter is the name of the file containing the reduction definitions for your database. The -r option must be used with the -l or -u option.

-s specifies the file that **tqzip** should use for a derived statistics load or unload operation. The *derived-file* parameter is the name of the file containing the derived statistic definitions for your database. The -s option must be used with the -l or -u option. See 8.4.1 and 8.4.2.

-t specifies the file that **tqzip** should use for a threshold load or unload operation. The *threshold-file* parameter is the name of the file containing the threshold definitions for your database. The -t option must be used with the -l or -u option. See 7.13 and 7.14.

-w specifies the file that **tqzip** should use for a workload load or unload operation. The *workload-file* parameter is the name of the file containing the workload definitions for your database. The -w option must be used with the -l or -u option. See 5.4.1 and 5.4.2.

Appendix I
2200 Socket Interface

TeamQuest Baseline uses the TCP/IP interface supplied by the Communications Application Program Interface to provide the capability of connecting from TeamQuest View or TeamQuest Performance Software Administration to **tqdata** using an OS 2200 system socket. To use this connection option, you need to

1. Install the Communications Application Program Interface product and make the related changes to your CMS 1100 or CPComm configuration. Then, start the Communications Application Program Interface background run on your OS 2200 system.
2. If you use an encrypted user-id or password in the TQTCP configuration element, you must have the Unisys Cipher Application Programming Interface (API) product installed on your system. See I.2.3 and I.2.5.1 for additional information.
3. Update the SYSLIB*TQ-BASELINE.TQTCP-CONFIG element with the proper configuration information for your site. Then, start the TQTCP background run on your OS 2200 system.
4. Open a socket connection to your OS 2200 system from TeamQuest View or TeamQuest Performance Software Administration.

I.1. Communications Application Program Interface

Communications Application Program Interface is a Unisys product that is bundled with releases HMP IX 4.0 and later. It is an extended mode subsystem and a background run that provides an application interface to a communications network. Access to the network is made through the TSAM interface of CMS 1100 or the API interface of CPComm using either TCP/IP or UDP protocols. Communications Application Program Interface gives an application program the ability to access the network by using simple commands such as LISTEN, CONNECT, SEND, RECEIVE, and DISCONNECT. A background run performs the basic network control functionality through CMS 1100 and CPComm.

Both the Communications Application Program Interface subsystem and background run are automatically installed using SOLAR. You must also create a data file called COMAPI*CONFIG that contains the appropriate Communications Application Program Interface configuration entries as described in the *ClearPath Enterprise Servers Communications Application Program Interface (COMAPI) User's Guide*. In addition, a PROCESS statement for Communications Application Program Interface must be added to your CMS 1100 or CPComm configurations. The Communications Application Program Interface background run must be active on your system whenever you are using a socket connection to your OS 2200 system through TeamQuest View or TeamQuest Performance Software Administration.

For more information on how to install and configure the Communications Application Program Interface, update your CMS 1100 or CPComm configuration for the Communications Application Program Interface, and manage the Communications Application Program Interface background run, see the *ClearPath Enterprise Servers Communications Application Program Interface (COMAPI) User's Guide* that accompanies your ClearPath OS 2200 release documents.

I.2. TQTCP

The TeamQuest Baseline includes a TQTCP executable and background run. The TQTCP background run uses the Communications Application Program Interface to listen for incoming connections from TeamQuest View or TeamQuest Performance Software Administration. When a connection is made, TQTCP passes the TeamQuest View or TeamQuest Performance Software Administration conversation to a **tqdata** session that is executing in a new RSI$ run.

Both the TQTCP executable and background runs are installed with TeamQuest Baseline. You must also update the configuration element in SYSLIB*TQ-BASELINE.TQTCP-CONFIG before starting the TQTCP background run. The TQTCP background run must be active on your system whenever you are using a socket connection to your 2200 system through TeamQuest View or TeamQuest Performance Software Administration.

I.2.1. Background Run

The TQTCP runstream is installed into SYSLIB*RUN$. You can start the TQTCP background run by entering the following keyin at the system console:

```
ST TQTCP
```

To start the background run from a demand or batch run, use the following statement:

```
@start sys$lib$*run$.tqtcp
```

By default, the TQTCP runstream executes TQTCP without processor call options. If you need to use any options, you will need to update the TQTCP runstream accordingly before it is started.

The background run provides a keyin interface for connection status reporting and TQTCP termination. By default, the keyin name is TQTCP. You can use the -k option on the TQTCP processor call to change this keyin name if needed. A keyin is executed by entering the keyin name followed by one of these keyin values:

STATUS	provides information on open connections.
END	waits for open connections to terminate on their own before terminating TQTCP. No new connections are accepted after the END keyin has been received by TQTCP.
ABORT	terminates TQTCP aborting any open connections. Using ABORT may have unpredictable consequences during termination of the peer (tqview or tqadmin) end of the connection. The END keyin is the preferred method for terminating TQTCP.

Examples

```
TQTCP STATUS
```

Prints a list of open connections on the system console.

```
TQTCP END
```

Terminates TQTCP after open connections have closed.

```
TQTCP ABORT
```

Terminates TQTCP after aborting any open connections.

I.2.2. Processor Call

The TQTCP program has the following processor call format:

```
@TQTCP [-c, config-file.elt] [,-k, keyin-name] [,-r] [,-u]
```

where:

-c specifies the TQTCP configuration element. By default, the configuration element is located in SYS$LIB*TQ-BASELINE.TQTCP-CONFIG. At a minimum, values must be provided in this configuration element for host, user-id, password, and account.

-k specifies a keyin name for the TQTCP background run. By default, *keyin-name* is TQTCP.

-r runs TQTCP in real-time mode. Each connection made to the socket will operate in real-time mode. When using the -r option, both the account for the TQTCP background run and the account on the *account* statement in the TQTCP configuration element must allow real-time mode.

-u specifies to use unique user-ids for each user's TQTCP run and controls which users can run TQ-View or TQ-Administration.

I.2.3. Configuration Element

You supply information in the TQTCP configuration element, SYSLIB*TQ-BASELINE.TQTCP-CONFIG, that is used by the master TQTCP run to start subsequent TQTCP runs associated with each new socket connection. When a user connects using a socket through TQ-View or TQ-Administration, a new TQTCP run is initiated for each session. Each statement in the TQTCP configuration element contains a label and a value, separated by a space. The "#" can be used to indicate a comment line.

There are two different formats of information that can be entered in the TQTCP configuration element. If you prefer that all TQTCP runs are started with the same user-id, password, account, and so on, see I.2.3.1.

If you prefer that each TQTCP run be identifiable to the user-id that started it, see I.2.3.2. Using this format, enabled by using the -u option on the TQTCP call line, will restrict which user's can use TQ-View or TQ-Administration in accessing the TeamQuest Performance database. Both TQ-View and TQ-Administration solicits a user-id and password to be used in initiating the associated TQTCP run. If the user-id entered exists in the TQTCP configuration element, then that user is allowed access. The TQTCP run will be successfully started if the user-id, password, and account are valid in signing on to the 2200 system. When the tqinit is performed for TQ-Baseline or TQ-Online, you are asked if you prefer unique user-ids for socket connections.

I.2.3.1. Same User-id For All TQTCP Runs

The user-id, password, and account configuration statements are required. All other configuration statements are optional and can be used when the defaults are not adequate.

The following configuration statements are allowed.

interface *interface-type*
: This tells TQTCP the communications interface to use when listening for incoming connections. The interface type can be either CMS1100 or CPCOMM. If this statement is not supplied, the default is CPCOMM.

port *port-number*
: This tells TQTCP the port number to listen on for incoming connections. If this statement is not supplied, the default port number is 2700.

userid *xxxxx*
password *xxxxx*
: The user-id and password values are used to initiate the RSI$ run that is created to handle a socket connection. These statements are required and standard OS 2200 rules must be followed for each *xxxxx* value. The user-id must have the security attribute specifying that the @RUN image is always user-entered or @RUN image is user-entered or system-generated. Also, terminal timeout must be disabled for the user-id. If you need to use an encrypted user-id and password, use the euser and epswd statements.

euser *xxxxx*
epswd *xxxxx*
: An encrypted userid (euser) and password (epswd) that can be substituted for the userid and password. See I.2.5.1 for information on using an encrypted userid and password.

runid *xxxxx*
account *xxxxx*
project *xxxxx*
: The run-id, account, and project values are used on the @RUN statement for the RSI$ run that is created to handle a socket connection. Standard OS 2200 rules must be followed for each *xxxxx* value. Account may also specify a user-id for the @RUN by entering account/userid. The account statement is required. If running TQTCP in real-time, this account must allow real-time mode. Otherwise, the RSI$ runs started to handle each connection will abort immediately. Runid defaults to TQTCP, and project defaults to TQDATA$ if they are not specified.

siteid *xxxxx*

Optional list of site-ids to use for RSI$ runs. If not present, site-ids will automatically be generated by RSI$. Site-ids are specified one per line. Up to 50 concurrent RSI$ runs can be executing within TQTCP. Each socket connection initiates an RSI$ run. If using specific site-ids, TQTCP will use the first 50 site-ids specified here. If there are more, the additional ones are ignored. If there are fewer, the number of site-ids specified is used as the maximum number of concurrent RSI$ runs.

xqtfile *qualifier*filename.*

This file is where the **tqdata** executable resides. This file is used in the RSI$ run to execute **tqdata**. If this statement is not supplied, the default is SYSLIB*TQ-BASELINE.

The following is an example of a TQTCP configuration element:

```
interface CPCOMM
port 2700
userid tquser
password 123456
runid tqrun
account 11000
project tqproj
siteid TQ001
siteid TQ002
siteid TQ003
siteid TQ004
siteid TQ005
xqtfile tq*file.
```

Note: *Remember that the only required statements are userid and account. If the defaults for the other configuration statements are acceptable, you do not need to include any of the optional configuration statements in your TQTCP configuration element.*

In this example, TQTCP would use the CPCOMM interface to listen on port 2700 for incoming connections from TeamQuest View or TeamQuest Performance Software Administration. When a connection is made, an RSI$ terminal would be initiated using one of the five specified site-ids (TQ001 through TQ005). The RSI$ run would be started using the following user-id, password, and @RUN statement:

```
*tquser/123456
@run tqrun,11000,tqproj
```

The following processor call statement would be used to execute **tqdata** in the RSI$ run (*socketid* is inserted by TQTCP):

```
@tq*file.tqdata -c,socketid
```

Since five site-ids were specified in this example, TQTCP would allow up to five connections operating concurrently and would use the specified site-ids (TQ001 through TQ005) when initiating RSI$ runs to handle each connection. If no site-id statements were included in the TQTCP configuration element, up to 50 concurrent connections would be allowed and RSI$ would automatically generate the site-ids for each connection.

I.2.3.2. Unique User-ids For Each TQTCP Run

The user-id and account are the only statements that are required. The password used to start the TQTCP run is solicited by TQ-View or TQ-Administration and should not be contained in the TQTCP configuration element. All user-ids that are allowed to run either TQ-View or TQ-Administration must exist in the TQTCP configuration element.

interface *interface-type*
: This tells TQTCP the communications interface to use when listening for incoming connections. The interface type can be either CMS1100 or CPCOMM. If this statement is not supplied, the default is CPCOMM.

port *port-number*
: This tells TQTCP the port number to listen on for incoming connections. If this statement is not supplied, the default port number is 2700.

siteid *xxxxx*
: Optional list of site-ids to use for RSI$ runs. If not present, site-ids will automatically be generated by RSI$. Site-ids are specified one per line. Up to 50 concurrent RSI$ runs can be executing within TQTCP. Each socket connection initiates an RSI$ run. If using specific site-ids, TQTCP will use the first 50 site-ids specified here. If there are more, the additional ones are ignored. If there are fewer, the number of site-ids specified is used as the maximum number of concurrent RSI$ runs.

xqtfile *qualifier*filename.*
: This file is where the **tqdata** executable resides. This file is used in the RSI$ run to execute **tqdata**. If this statement is not supplied, the default is SYSLIB*TQ-BASELINE.

The following information should be entered for each user allowed to run TQ-View or TQ-Administration:

userid *xxxxx*
: The user-id and password values are used to initiate the RSI$ run that is created to handle a socket connection. These statements are required and standard OS 2200 rules must be followed for each *xxxxx* value. The user-id must have the security attribute specifying that the @RUN image is always user-entered or @RUN image is user-entered or system-generated. Also, terminal timeout must be disabled for the user-id. If you need to use an encrypted user-id and password, use the euser and epswd statements.

euser *xxxxx*
: An encrypted user-id (euser) that can be substituted for the user-id. See I.2.5.1 for information on using an encrypted user-id.

runid *xxxxx*
account *xxxxx*
project *xxxxx*
: The run-id, account, and project values are used on the @RUN statement for the RSI$ run that is created to handle a socket connection. Standard OS 2200 rules must be followed for each *xxxxx* value. Account may also specify a user-id for the @RUN by entering account/user-id. The account statement is required. If running TQTCP in real-time, this account must allow real-time mode. Otherwise, the RSI$ runs started to handle each connection will abort immediately. If not supplied, the run-id defaults to the first 6 characters of the user-id and the project defaults to TQDATA$.

The following is an example of a TQTCP configuration element using unique user-ids:

```
interface CPCOMM
port  2700
# comment line - include information for each user
userid  john
runid john
account 11000
project tqproj
#  comment line - an encrypted user-id for Sally
euser 0fa3ca0a57fb6478f96e815528a4fb340614
account 11000
project tqproj
```

Note: *Remember that the only required statements are userid and account. If the defaults for the other configuration statements are acceptable, you do not need to include any of the optional configuration statements in your TQTCP configuration element.*

In this example, only userid “john” and “sally” are allowed to run TQ-View or TQ-Adminstration. When a connection is made, an RSI$ run would be started using the associated information above and the password solicited from either TQ-View or TQ-Administration. The RSI$ runs for john and sally would look like the following:

```
*john/password
@run john,11000,tqproj

*sally/password
@run sally,1100,tqproj
```

I.2.4. Error Messages

If the TQTCP background run terminates prematurely, error information will be available in the TQDATA$*TQTCP$PRT file. Communications Application Program Interface errors are output by TQTCP as follows:

Error COMAPI returned (*status*/*aux*) on TCP_*xxxxx* for connection <*connection-id*>

where:

status
: is the Communications Application Program Interface error status and aux is the Communications Application Program Interface auxiliary status. See the *ClearPath Enterprise Servers Communications Application Program Interface (COMAPI) User’s Guide* for a description of these error statuses.

TCP_*xxxxx*
: is the name of the Communications Application Program Interface function that was called when the error occurred (for example, TCP_LISTEN, TCP_SEND, or TCP_RECEIVE).

connection_id
: is the session identifier obtained from the Communications Application Program Interface for the TCP session that encountered the error.

I.2.5. Security Considerations

Since the TQTCP configuration element contains a valid user-id, password, and account, you may want to limit access to this information. You can use OS 2200 file security to limit access to the configuration element. To do this:

1. Catalog a new file and copy SYSLIB*TQ-BASELINE.TQTCP-CONFIG to it.
2. Attach the appropriate security to this new file using SIMAN.
3. Update the new configuration element to include, at a minimum, the user-id, password, and account configuration statements.
4. Update the TQTCP processor call in the TQTCP runstream to include the -c option to specify the location of the secured configuration element.

Remember that, at a minimum, the TQTCP run must be able to read the configuration element.

I.2.5.1. User-id and Password Encryption

A FIPS 140-2 compliant encrypted user-id and password can also be created. The TQENC program is used for encrypting the user-id and password used in the TQTCP configuration element. To use this program, you must have the Unisys Cipher API product installed on your system. Cipher API is governed by the Federal Information Processing Standards (FIPS) Publication 140-2, *Security Requirements for Cryptographic Modules*.

When using the TQENC program, you are prompted for a user-id and password to encrypt. The TQENC program then displays the configuration statements that can be cut and pasted into the TQTCP configuration element described in I.2.3.

The following shows an example of using the TQENC program:

```
>@TQENC
>Copyright (c) 1992-2012 TeamQuest Corporation. All Rights Reserved.
>
>Enter a user-id to encrypt (user-id, <none>):
>userid1
>
>Enter a password to encrypt (password, <none>):
>password1
>
>Include the following lines in your TQTCP configuration element:
>
>euser 0da184c595a3acd48b669b1520c760057017
>epswd 639692a250c9e94a383a2f952e449b7468ec
```

I.3. Connecting to the Socket

To access your OS 2200 system using a socket connection, you must use TeamQuest View or TeamQuest Performance Software Administration. The default port for socket connections is 2700.

After you launch TeamQuest View or TeamQuest Performance Software Administration, use the following steps to define a new session:

1. Select File > Session > New from the menu bar
2. In the New Session dialog box, specify
   - Socket as the connection type
   - The Host Name or IP address of your OS 2200 system
   - The Port number in use by TQTCP on your OS 2200 system

When you click OK in the New Session dialog box, the connection is made to your OS 2200 system and a TeamQuest View or a TeamQuest Performance Software Administration session is initialized.

Appendix J
2200 SNMP Trap Alarms

J.1. Overview

The TeamQuest Performance Software provides the capability of sending a Simple Network Management Protocol (SNMP) trap as an alarm action. Using the **tqzip** program, you can define thresholds that will send an alarm event to any management console that accepts SNMP traps. Most management consoles allow you to specify a variety of actions upon receipt of an SNMP trap such as making an entry in an event window, popping up an alarm box window, or even running a program such as TeamQuest View.

The TeamQuest Baseline uses the SNMP TSAM interface supplied by the Unisys product CMS 1100 to send SNMP traps when an alarm occurs on a managed node. The **tqalm** program of TeamQuest Baseline continuously runs to monitor for alarm conditions on the system. In CMS 1100 network management terms, **tqalm** is an SNMP Management Agent Extension that sends SNMP traps to the CMS SNMP agent. The CMS SNMP agent then forwards the traps to network manager(s) that are listening on port 162 on the management host(s).

To send SNMP traps for alarms, you need to:

1. Update your CMS 1100 configuration to define the TQSSU process and the trap destinations (see J.2).
2. Create TeamQuest Baseline threshold definitions that contain the SNMPTrap command in one of the user text fields (see J.3).
3. Update your network manager configuration to allow it to recognize and accept TeamQuest Performance Software SNMP traps (see J.5).
4. Ensure that threshold checking is turned on in TeamQuest Baseline and that the **tqalm** run is executing on your system. The TeamQuest data collection probes that continuously execute on your system must also be up and running.

J.2. CMS 1100 Configuration

To use the SNMP TSAM interface for SNMP traps, you need to add a PROCESS statement for TQSSU to your CMS 1100 configuration. This statement defines a process that allows **tqalm** to initialize itself as an SNMP service user (SSU). The following PROCESS statement must be used:

```
PROCESS,TQSSU    TYPE,TSAM    PASSWORD,TEAMQUEST
```

To define the destination systems for your SNMP traps, use the SNMP-MGMT statement in your CMS 1100 configuration. This statement defines the community name and host addresses that are to receive the traps. For example:

```
SNMP-MGMT,MANAGE    COMMUNITY-NAME,public  WRITE-ACCESS,NO  ;
                    TRAPS,public,xxx.xxx.xxx.xxx  ;
                    TRAPS,public,xxx.xxx.xxx.xxx
```

For more information on how to update your CMS 1100 configuration for using SNMP traps, see the *OS 2200 Communications Management System (CMS 1100) Installation and Configuration Guide.* For information on the PROCESS and SNMP-MGMT statements, see the *OS 2200 Communications Management System (CMS 1100) Configuration Reference*.

Note: *The Unisys product, CPComm, also supports the SNMP TSAM interface required for sending SNMP traps. If you are using CPComm, you will need to make the appropriate configuration changes for defining the TeamQuest SSU and trap destinations.*

J.3. TeamQuest Alarm Administration Interface

This feature is implemented as part of the Alarms and Thresholds capability of the TeamQuest Performance Software (see Section 7). When a threshold is defined using the **tqzip** program, you can specify an SNMP trap action by placing a special syntax in one of the user text fields of the threshold definition.

When an alarm is raised and this special syntax is present, an SNMP trap is sent according to the specifications in the syntax. Specifically, an SNMPv1 trap is sent with a generic trap type of “enterprise specific (6)” and a specific trap type that is used to further identify the contents of the trap (see J.4.1 for a list of the specific trap types).

Information about the alarm is placed in the “variable bindings” section of the trap and is available for display on a management console or any other use. There are 11 fields of data that describe the alarm and the default is to send all of these fields in the trap. However, you can also choose to send only selected fields depending on your site requirements.

SNMP Trap Syntax

You specify an SNMP trap alarm action by placing a special syntax on any one or more of the User Text fields in a threshold definition (see 7.4). The syntax for this command is as follows:

```
SNMPTrap  [Fields=field list]
```

The *field list* is an optional list of trap variables to send to the trap destination. It consists of any of the variable names described in J.4.2. The variable names in the list must be separated by a comma (,). If the Fields parameter is not specified on the command, the default is to send all variables.

The following rules apply when entering the SNMP trap syntax on the User Text fields in a threshold definition:

- The SNMPTrap keyword must begin in the first character position of the field.
- The SNMPTrap and Fields keywords are case-sensitive and must be separated by one or more spaces.
- The command can be continued to the next User Text field by placing a backslash (\) as the last entry on the image. Each User Text field can be up to 60 characters long.
- Only one SNMPTrap action can be specified for each threshold. The User Text fields that contain this command will not be part of an alarm message sent to the console or become part of the trap's fields.

Example 1

The following example shows the most common form of an SNMP trap alarm. It sends a trap containing all fields to port 162 of the host and community name configured in CMS 1100.

```
SNMPTrap
```

Example 2

This command limits the fields sent in the trap. Only System, Identifier, Value, and ThreshZones are included in the trap.

```
SNMPTrap  Fields=System,Identifier,Value,ThreshZones
```

Note: *If you can format the event message derived from the trap at the management console, you should probably send all trap fields. The capability to send selected trap fields is provided for those situations where little or no control is available over the event message displayed on the console. In those cases, you may want to send only selected fields to limit message content or size.*

J.4. SNMP Trap Contents

The contents of the SNMP trap that is sent to the management console is an SNMPv1 trap protocol data unit (PDU). It contains a set of fixed fields and an optional set of variable fields called variable bindings or varbinds. The mapping of the trap fields is described in J.4.1 and J.4.2.

J.4.1. SNMP Trap Mappings

The following fields are present in every SNMPv1 trap PDU and are mapped as follows:

Enterprise
: This field is set to the enterprise identifier for Unisys, OS2200, TeamQuest, which is {1, 3, 6, 1, 4, 1, 223, 7, 6}.

Agent Address
: This field is set to the IP address of the host on which the alarm occurred. The symbolic host name is also available in the System field of the variable fields (see J.4.2).

Generic Trap
: This field is set to the enterprise specific (6) trap type in all cases.

Specific Trap
: This value is used to further identify the contents of the trap.

  1 TeamQuest Performance Software alarm with a severity of Normal (or Harmless)

  2 TeamQuest Performance Software alarm with a severity of Warning.

  3 TeamQuest Performance Software alarm with a severity of Critical.

Timestamp
: This field is set to sysUpTime, which is the amount of time since the host was started.

J.4.2. Variable Fields

The following fields are specific to the TeamQuest Baseline SNMP trap and are defined formally in the TeamQuest Enterprise Management Information Base (MIB). See J.6 for more information on the MIB. These variable fields are mapped as follows:

System
: This field is set to the name of the host that generated the alarm.

Identifier
: This field is set to the Alarm Identifier value from the threshold definition and should contain a description of the entity or variable that raises the alarm.

Severity
: Severity is based on the difference between the current and normal zone values.

  In all situations, the severity is NORMAL if the current and normal zone match.

  In situations where 2 zones are defined, the zone difference will be either 0 or 1. If the difference is 1, the severity is CRITICAL.

  In situations where 3 zones are defined, the zone difference is either 0 or 1 or 2. If the difference is 1 and the normal zone is 1 or 3, the severity is WARNING. If the difference is 1 and the normal zone is 2, the severity is CRITICAL. If the difference is 2 and the normal zone is 1 or 3, the severity is CRITICAL.

Value
: This is set to an ASCII representation of the value that triggered the alarm.

ThreshZones
: This is set to a character string that represents the threshold's zones and the normal, current, and previous positions within those zones. The following is the format of this string:

  ```
  ----<thresh1>----<thresh2>----
  ```

  where <*threshn*> is the threshold value and ---- represents any indicators for that zone at the time of the alarm. The possible indicators are Norm, Curr, and Prev.

  For example,

  ```
  Norm<75.00>Curr<95.00>Prev
  ```

  This string indicates a warning(2) situation where the parameter is currently in the second zone, which is not the normal zone, and was previously in a critical(3) state.

DateTime
: This is set to an ASCII representation of the date and time that the alarm was raised. For example,

```
01/13/2012 02:45:00
```

Text1
: This is set to the contents of User Text 1 if that field did not contain this SNMPTrap command.

Text2
: This is set to the contents of User Text 2 if that field did not contain this SNMPTrap command.

Text3
: This is set to the contents of User Text 3 if that field did not contain this SNMPTrap command.

Database
: This is set to the name of the database that contains the data used to detect the alarm condition. This field can be used to open a report for a specified parameter using the TeamQuest View command line.

Parameter
: This is set to the parameter name of the entity that raised the alarm. This field can be used to open a report for the specified parameter using the TeamQuest View command line.

These TeamQuest Baseline specific fields are part of the TeamQuest Enterprise MIB (see J.6). This MIB is supplied with TeamQuest Baseline in text form so that it can be loaded into a management console if desired. The MIB defines a group called tq2200AlarmTrap that contains the variable trap fields described above. The field names in the MIB have “tq2200Alarm” as a prefix on the field names.

J.5. Management Console Configurations

The TeamQuest Baseline comes with configuration information for the Hewlett-Packard OpenView product and the Tivoli Enterprise Console (TEC) product.

The configuration information supplied for the HP OpenView product requires that, at a minimum, the System, Identifier, Severity, Value, and ThreshZone trap variables be sent with the TeamQuest Performance Software alarm trap. For the TEC product, the System, Identifier, and Value trap variables must be sent with the trap. You can use the field list on the SNMPTrap command to specify which trap variables are to be sent. See J.4.2 for a list of the trap variables.

By default, all trap variables are sent with each alarm trap. However, only the minimum set of trap variables is displayed at the management console. To have the additional trap variables displayed, you must update the configuration information for your management console product.

J.5.1. HP OpenView Configuration

To make your TeamQuest Performance Software alarm display a message on an HP OpenView Event Browser window, you must first define a threshold using the **tqzip** program and then configure the HP OpenView trap daemon to accept and process the trap. Follow these steps:

1. Define a threshold using the **tqzip** program and add the SNMPTrap command to one of the User Text fields in the threshold definition. Since you can select fields in the HP OpenView trapd configuration, you do not need to specify which fields to send on the SNMPTrap command.

   On the SNMP-MGMT statement in the CMS 1100 configuration, you must specify the host that has the HP OpenView trap daemon (trapd or ovtrapd) running on it. Also, the trap daemon must be listening on port 162, which is the destination port for all CMS 1100 SNMP traps.

2. Configure the HP OpenView trap daemon (trapd or ovtrapd) to accept and process your alarm trap. To do this, merge the appropriate TeamQuest-supplied file into the trapd.conf file as documented in the trapd man pages. These entries define the (Unisys, OS2200, TeamQuest) enterprise number to trapd and format a message for the event browser (xnmevents) for each trap type. Additionally, a pop-up window is displayed for critical alarm traps.

   Two symbolic elements have been supplied to assist you in updating the HP OpenView trapd.conf file. These elements are located in the file, SYSLIB*TQ-BASELINE. The element names are OV3TRAPD and OV4TRAPD. You should use the element that corresponds to your level of HP OpenView (for example, HP OpenView level 3 users would use OV3TRAPD and HP OpenView level 4 or 5 users would use OV4TRAPD).

3. When you have updated the trapd.conf file, you must follow the HP OpenView procedures for restarting the HP OpenView trap daemon.

J.5.2. Tivoli Management Environment Configuration

To make your TeamQuest Performance Software alarm display a message on a TEC Event window, you must first define a threshold using the **tqzip** program, configure the TEC SNMP Event Adapter to accept and process the alarm trap, and define the TeamQuest event classes to the TEC Event Server. Follow these steps:

1. Define a threshold using the **tqzip** program and add the SNMPTrap command to one of the User Text fields in the threshold definition. Since you can select fields in the TEC SNMP Event Adapter configuration, you do not need to specify which fields to send on the SNMPTrap command.

   On the SNMP-MGMT statement in the CMS 1100 configuration, you must specify the host that has the TEC SNMP Event Adapter (tecad_snm or tecad_snmp) running on it. Also, the TEC SNMP Event Adapter must be listening on port 162, which is the destination port for all CMS 1100 SNMP traps.

2. Configure the TEC SNMP Event Adapter to accept and process your alarm trap. You can easily tailor the TEC SNMP Event Adapter to format a suitable Tivoli event from your trap and forward this event to an event server. The event server then displays this event on the appropriate event console. To configure the TEC SNMP Event Adapter, you must update three configuration files as described in steps 3 through 5.

3. Merge the following entries into the tecad_snmp.oid file. These entries should be added to the file so that the file remains in increasing object identifier order. The following entries have been supplied to you in the symbolic element, SYSLIB*TQ-BASELINE.TECAD-OID. You can use this file to update the SNMP Event Adapter tecad_snmp.oid file.

```
# TeamQuest 2200 MIB
# "enterprises"              "1.3.6.1.4.1"
# "unisys"                   "1.3.6.1.4.1.223"
# "os2200"                   "1.3.6.1.4.1.223.7"
# "tq2200"                   "1.3.6.1.4.1.223.7.6"
# "tq2200AlarmTrap"          "1.3.6.1.4.1.223.7.6.1"
"tq2200AlarmSystem"          "1.3.6.1.4.1.223.7.6.1.1"
"tq2200AlarmIdentifier"      "1.3.6.1.4.1.223.7.6.1.2"
"tq2200AlarmSeverity"        "1.3.6.1.4.1.223.7.6.1.3"
"tq2200AlarmValue"           "1.3.6.1.4.1.223.7.6.1.4"
"tq2200AlarmThreshZones"     "1.3.6.1.4.1.223.7.6.1.5"
"tq2200AlarmDateTime"        "1.3.6.1.4.1.223.7.6.1.6"
"tq2200AlarmText1"           "1.3.6.1.4.1.223.7.6.1.7"
"tq2200AlarmText2"           "1.3.6.1.4.1.223.7.6.1.8"
"tq2200AlarmText3"           "1.3.6.1.4.1.223.7.6.1.9"
"tq2200AlarmDatabase"        "1.3.6.1.4.1.223.7.6.1.10"
"tq2200AlarmParameter"       "1.3.6.1.4.1.223.7.6.1.11"
```

4. Update the tecad_snmp.cds file in use by the SNMP Event Adapter by appending the entries contained in the element SYSLIB*TQ-BASELINE.TECAD-CDS. The tecad_snmp.cds file describes the mapping between SNMP traps and the events that are forwarded to the TEC Event Server.

5. Update the tecad_snmp.baroc file for the rule base that is currently in use by the TEC Event Server or import the TeamQuest class definitions into your rule base. You can add the TeamQuest class definitions to the rule base by appending the entries contained in SYSLIB*TQ-BASELINE.TECAD-BAROC to the tecad_snmp.baroc file for the rule base that is currently in use by the TEC Event Server.

   Another way to add the TeamQuest class definitions to the rule base is to import them using the TME desktop GUI or the wimprbclass command.

6. The last step is to do a warm start of the SNMP Event Adapter and to stop and restart the Event Server. Doing a warm start of the SNMP Event Adapter causes it to re-read its configuration files. These include the tecad_snmp.cds and tecad_snmp.oid files that were updated in steps 3 and 4. Stopping and restarting the Event Server allows it to read in the new event classes that were added to the rule base in step 5. The Event Server can be stopped and restarted from the TME desktop GUI or you can use the wstopesvr and wstartesvr commands.

J.5.3. Computer Associates Unicenter TNG

The TeamQuest Performance Software can also send SNMP traps for alarms to the Computer Associates Unicenter TNG Event Console. However, no predefined message definitions or automated actions are provided as part of this release.

J.6. TeamQuest MIB Definition

The following table lists the TeamQuest MIB containing the object identifiers for the enterprise specific variables in the alarm trap. This definition is also contained in SYSLIB*TQ-BASELINE.TQMIB-TXT. The entries in this element can be loaded into a management console to provide textual names for the trap variables, if desired.

```
--
-- Copyright (c) 2012 TeamQuest Corporation
-- All Rights Reserved
--
-- No use of this software is permitted except in accordance with
-- the governing license agreement and applicable copyright, patent
-- and trade secret laws.

-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
--
-- This file contains the ASN.1 source describing the TeamQuest Corporation
-- enterprise MIB for OS2200.  The enterprise MIB described is subject to
-- change in a future release, and support for any object described in this
-- MIB may be removed in a future release, as standard MIBs in the management
-- domain described by this MIB become defined and published as Internet RFCs.
--

TQ2200-MIB DEFINITIONS  ::= BEGIN

IMPORTS

      Enterprises                       FROM RFC1155-SMI
      OBJECT-TYPE                       FROM RFC-1212
      DisplayString                     FROM RFC1213-MIB
      TRAP-TYPE                         FROM RFC-1215;

--    TeamQuest enterprise

      tq2200                 OBJECT IDENTIFIER ::=
                             { enterprises unisys(223) os2200(7) 6 }

--    TeamQuest enterprise groups.  The only group defined initially
--    is the TeamQuest Alarm Trap group.  Other groups will be added as
--    required.

      tq2200AlarmTrap        OBJECT IDENTIFIER ::= { tq2200 1 }

--    The tqAlarmTrap group.  This group contains objects describing
--    the TeamQuest SNMP Alarm Trap.
```

```
tq2200AlarmSystem OBJECT-TYPE
   SYNTAX DisplayString (SIZE (0..52))
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
   "The alarm system is a textual string that identifies the system
    on which the alarm was raised."
    ::= { tq2200AlarmTrap 1 }

tq2200AlarmIdentifier OBJECT-TYPE
   SYNTAX DisplayString (SIZE (0..244))
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
   "The alarm identifier is a textual string that identifies the
    parameter for this alarm. The qualifier1, qualifier2, and
    qualifier3 fields are appended to this field if present."
    ::= { tq2200AlarmTrap 2 }

tq2200AlarmSeverity OBJECT-TYPE
   SYNTAX INTEGER {
     normal(1),
     warning(2),
     critical(3)
     }
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
   "The severity of this alarm as determined by the change in zones."
   ::= { tq2200AlarmTrap 3 }

tq2200AlarmValue OBJECT-TYPE
   SYNTAX DisplayString (SIZE (0..24))
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
   "The sampled value that caused the alarm to be raised.  This value
    is sent in display since we do not have a double type encoding for
    SNMP."
   ::= { tq2200AlarmTrap 4 }

tq2200AlarmThreshZones OBJECT-TYPE
   SYNTAX DisplayString (SIZE (0..80))
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
   "This string depicts the theshold zones along with their boundaries
    and indicates normal and current. The format of the field is
           ----<value1>----<value2>----
    Where ---- represents Norm, Curr, or Prev.  Example:
           Norm<45.00>Prev<75.00>Curr "
   ::= { tq2200AlarmTrap 5 }

tq2200AlarmDateTime  OBJECT-TYPE
   SYNTAX    DisplayString (SIZE(26..120))
   ACCESS    read-only
   STATUS    mandatory
   DESCRIPTION
   "The date and time that the alarm was generated in the format:
    MM/DD/YYYY HH:MM:SS (eg: '01/13/2012 02:45:00')."
   ::= { tq2200AlarmTrap 6 }
```

```
tq2200AlarmText1  OBJECT-TYPE
   SYNTAX    DisplayString (SIZE(0..64))
   ACCESS    read-only
   STATUS    mandatory
   DESCRIPTION
   "User text field 1"
   ::= { tq2200AlarmTrap 7 }

tq2200AlarmText2  OBJECT-TYPE
   SYNTAX    DisplayString (SIZE(0..64))
   ACCESS    read-only
   STATUS    mandatory
   DESCRIPTION
   "User text field 2"
   ::= { tq2200AlarmTrap 8 }

tq2200AlarmText3  OBJECT-TYPE
   SYNTAX    DisplayString (SIZE(0..64))
   ACCESS    read-only
   STATUS    mandatory
   DESCRIPTION
   "User text field 3"
   ::= { tq2200AlarmTrap 9 }

tq2200AlarmDatabase  OBJECT-TYPE
   SYNTAX    DisplayString (SIZE(0..256))
   ACCESS    read-only
   STATUS    mandatory
   DESCRIPTION
   "The current database name."
   ::= { tq2200AlarmTrap 10 }

tq2200AlarmParameter  OBJECT-TYPE
   SYNTAX    DisplayString (SIZE(0..256))
   ACCESS    read-only
   STATUS    mandatory
   DESCRIPTION
   "The current parameter string."
   ::= { tq2200AlarmTrap 11 }

END
```

J.7. Special Considerations

The TeamQuest Performance Software sends SNMP traps to the CMS 1100 SNMP agent. The CMS agent actually sends the traps to the network managers. The trap destinations are configured in CMS 1100. You specify the community name and host addresses where the trap is to be sent. The SNMP trap port number used by CMS 1100 is always 162. There is no way to change this port number in CMS 1100.

The length of the trap variable bindings is limited to 400 bytes by CMS 1100 (the limit is 128 bytes prior to CMS 8R3A). To stay within the byte limit, variable fields may be removed from the trap by the TeamQuest Performance Software before the trap is sent to CMS 1100.

The TeamQuest Performance Software runs on a variety of UNIX/Linux platforms, Microsoft Windows systems, and OS 2200 systems. Due to CMS 1100 limitations, the TeamQuest SNMP traps on OS 2200 systems use a different enterprise identifier and different specific trap values than the UNIX and Microsoft Windows systems TeamQuest traps. If you want to receive traps from these different types of systems at one network manager, you will need to make sure that your network manager configuration properly describes the OS 2200 system traps and the UNIX and Microsoft Windows systems traps. Using the network manager configuration elements supplied with the OS 2200 TeamQuest Performance Software and the configuration files supplied with the UNIX/Linux and Microsoft Windows systems TeamQuest Performance Software, you will be sure to have all of the SNMP trap formats properly defined to your network manager.

By default, **tqalm** initializes and terminates the TQSSU process in CMS 1100 each time a trap is sent. This results in the CMS 1100 slot being allocated for the shortest possible amount of time, but also causes the following console messages to be displayed by CMS 1100 each time a trap is sent:

```
PROCESS TQSSU IS UP
PROCESS TQSSU IS DOWN
```

If you would like to eliminate these console messages, update the SYSLIB*RUN$.TQALM runstream to use the -i option on the **tqalm** processor call statement:

```
@TQ$SYS.TQALM -c, -i
```

When the -i option is present, **tqalm** initializes the TQSSU process once at the start of execution. The TQSSU process will not be terminated until **tqalm** terminates. This eliminates the extra console messages, but ties up the CMS 1100 slot for the entire time that **tqalm** is running.

Glossary

A

aggregation set
A collection of parameter values, with each value representing the average for the parameter over a specified period of time.

ANSI
Abbreviation for American National Standards Institute.

C

capacity management
The process of managing computer system capacity for users to ensure that system processing can and always will deliver services to meet the user's productivity goals.

capacity planning
The function and activities concerned with determining future requirements for a computer system to meet the needs of the system users.

client-server model
A type of computer system architecture in which the components acting as the client make requests for data, services, or resources to those components acting as the server. The server performs the necessary operations to provide the requested data, services, or resources to the client.

Communications Application Program Interface (formerly called COMAPI)
Communications Application Program Interface is a combined extended mode subsystem and background run that allows applications to access a communications network.

correlation analysis
The process of comparing performance statistics to analyze system data.

D

data collection
: That portion of TeamQuest Baseline or TeamQuest Online host processing in which data for the performance database is collected from the data sources on the host system.

data display
: That portion of TeamQuest Baseline or TeamQuest Online processing in which TeamQuest View accesses data in the performance database and displays the data in graphic or tabular format.

E

ER
: *See* Executive request.

Exec
: *See* Executive.

Executive (Exec)
: A software program supplied by Unisys that controls the system operating environment. The Exec processes user runs, controls files, manages system resources, and performs input/output operations for the user. The Exec is the central component of the operating system and enforces security control.

Executive request (ER)
: A set of programmed instructions that let the Exec perform service functions such as reading and printing data and terminating a program. Selected ERs affect system security. The ability to execute these ERs is controlled, and this lets specified users override security restriction and validations.

G

granularity
: The time interval between points of data in the performance database.

H

history report
One of the TeamQuest Baseline and TeamQuest Online report types. History reports show performance data that has been stored in the performance database. They let you view past system performance and examine performance trends.

host software
The part of TeamQuest Baseline or TeamQuest Online that resides on the OS 2200 system and is responsible for collecting performance data, storing the data in the performance database, and maintaining the database. The host software is the server in the client-server model of the TeamQuest Performance Management Architecture.

host system
The system where the TeamQuest Baseline or TeamQuest Online software resides.

I

IOTRACE
An optional feature of SIP that can run independently or with SIP data collection. It gathers detailed information about every I/O operation to tape or mass storage.

M

monitor reports
One of the TeamQuest View report types. Monitor reports show real-time performance data as the activity occurs on the system.

P

parameter
A unique performance data item in the performance database.

parameter group
A section within a report definition file that identifies what parameter or parameters are retrieved from the host database. A report definition can contain multiple parameter group sections.

parameter value
A specific instance of data for a parameter in the performance database. A parameter value is identified by the parameter name, aggregation set, and time and date of the point.

performance evaluation
The analysis of system statistics to provide information on system activities and to identify any required corrective actions.

predefined reports
The set of reports that are part of the TeamQuest View software.

privilege
The mechanism of the OS 2200 security system that allows specified users to override mandatory and discretionary access controls.

probe
A component of the TeamQuest Performance Management Software that collects performance data from the operating system.

Q

Quota
The system resources that a run is allowed to use. The Exec can be configured to maintain Quota limits for each account number and user-id.

Quota set
A method of resource control used by the Exec. Quota sets limit the resources used by the attached account or user-id. You can attach Quota sets to accounts and user-ids. The Exec enforces the limits defined by the Quota set if the Quota level in the Exec is set.

S

SIP
(1) *See* Software Instrumentation Package. (2) The Quota privilege to allow all SIP functions. This privilege is required for the account running TeamQuest Baseline or TeamQuest Online on the host.

SIPONOFF
The Quota privilege to allow turning SIP on and off by using the ER SYSBAL$. This privilege is required for the account running TeamQuest Baseline or TeamQuest Online on the host.

SIPREDCT
The Quota privilege to allow SIP data reduction. This privilege is required for the account running TeamQuest Baseline or TeamQuest Online on the host.

Software Instrumentation Package (SIP)
An Exec system performance data collection function. SIP is configured during a system generation, and then activated or deactivated by the operator at the system console by using system balance keyins.

Software Library Administrator (SOLAR)
An OS 2200 software product used to load OS 2200 software products from a software package tape. You must use SOLAR to install TeamQuest Baseline and TeamQuest Online.

SOLAR
See Software Library Administrator.

system log
A file maintained by the Exec to keep historical record of the activity in the system. The system log stores, for example, the time a run starts and ends, console messages, checkpoints, and file information.

T

TeamQuest Model
A software product that lets you build and solve a network queuing model and thus predict the performance of your computer system. The **tqgetm** program of TeamQuest Baseline retrieves performance data that can be input to the OS 2200 Model Builder Wizard.

TeamQuest Performance Management Software
An open architecture designed around the client-server model and upon which all products in the TeamQuest Performance Management Software family are based.

TeamQuest View
The workstation software that provides the graphical user interface to the host software. TeamQuest View is the client in the client-server model of the TeamQuest Performance Management Software family.

TIP Performance Monitor (TPM)
TPM provides performance analysis of TIP/HVTIP transactions.

trend analysis
The process of identifying periodic oscillations, continuous growth, and continuous decline for the workloads on a computer system.

W

workload
The logical classification of work on a computer system.

X

XIIP

The XPC interface that connects an XPC to the system. One or more XIIPs can be used per XPC.

XPA

Refers to the Extended Processing Architecture of OS 2200 computer systems. This includes the 2200/500 and the 2200/900 systems.

XPC

Abbreviation for the Extended Processing Complex. This processing complex can be used to accelerate file accesses.

Bibliography

Business Information Server for ClearPath OS 2200 Administration Guide (7831 9415). Unisys Corporation.

ClearPath Enterprise Servers Communications Application Program Interface (COMAPI) User's Guide (7851 5749). Unisys Corporation.

ClearPath OS 2200 Cipher Application Programming Interface (API) Programming Reference Manual (3826 6110). Unisys Corporation.

ClearPath OS 2200 Exec System Software Administration Reference Manual (7831 0323). Unisys Corporation.

ClearPath OS 2200 Software Products Installation Guide (7831 0612). Unisys Corporation.

ClearPath OS 2200 TeamQuest PMLog End Use Reference Manual (TQ–01351). TeamQuest Corporation.

ClearPath OS 2200 TeamQuest Probes End Use Reference Manual (TQ–01361). TeamQuest Corporation.

ClearPath OS 2200 TeamQuest Products Installation Guide (TQ–01013). TeamQuest Corporation.

ClearPath OS 2200 Transaction Processing Administration and Operations Reference Manual (7830 7881). Unisys Corporation.

Distributed Processing Middleware Open Distributed Transaction Processing Administration Guide Volume 1: Installation, Configuration, and Ongoing Administration (7833 5072). Unisys Corporation.

OS 2200 Communications Management System (CMS 1100) Configuration Reference Manual (7830 9853). Unisys Corporation.

OS 2200 Communications Management System (CMS 1100) Installation and Configuration Guide (7830 9846). Unisys Corporation.

OS 2200 Exec System Software Operations Reference Manual (7831 0281). Unisys Corporation.

OS 2200 Linking System Subsystems Programming Guide (7830 7451). Unisys Corporation.

OS 2200 System Log Operations and Support Reference Manual (7831 0315). Unisys Corporation.

Single Point Autoaction Message System Administration Guide (7833 4760). Unisys Corporation.

SMART for ClearPath OS 2200 Administration and Operations Guide (7830 7337). Unisys Corporation.

TeamQuest Model User Guide for Unisys Systems (TQ–01424). TeamQuest Corporation.

TeamQuest View User Guide (TQ–01401). TeamQuest Corporation.

Index

A

abnormal termination
  of **tqbsp** 9–3
  of **tqsam** 9–7
adding
  collection period for tables 11–7
  retention specification for tables 11–13
Admin utility *See* TeamQuest Performance Software Administration
administration sessions, terminating 4–8
aggregation set
  compact 6–7
  consolidation 6–4
  converting 6–26
  creating 6–10, 6–16
  deleting 6–15
  description 1–10, 2–1
  displaying 6–14
  modifying 6–14
  multiple sets in a database 2–1
  structure 6–2
  tqasa administration 6–15
aggregation sets, description 3–3
alarm administration, SNMP traps J–2
alarm handling
  description 2–5, 7–1
  introduction 3–3
  user-defined 7–31
alarm messages
  description 7–30
  file of 7–27
  format 7–30
  viewing 7–27
alarms
  considerations for using TeamQuest Baseline with SPO 7–32
  handling 7–28
  maintaining 7–1
  severity indicator 7–1
Alarms and Thresholds Menu 7–5
alias
  defining 8–11
  deleting 8–13
  qualifying parameters 8–12
applying
  collection period changes 11–9
  retention specification changes 11–15

B

backup of performance database 10–8

C

capabilities of TeamQuest Baseline 1–5
checking thresholds 7–21
Client Table, Open Distributed Transaction Processing A–64
CMS 1100 configuration for SNMP traps J–2
Collection Period Menu of **tqzip** 11–4
collection periods
  administering 11–4
  description 11–1
Communications Application Program Interface overview I–1
condensed output of **tqrpt** D–4
connections
  named pipe 4–6
  socket 4–5
  SSL 4–6
contents of SNMP traps J–4
conventions, report definition file C–2
converting expanded aggregation sets 6–26
creating
  aggregation set 6–10, 6–16
  derived statistic 8–8
  derived statistic threshold definitions 7–14

threshold definitions 7–6
workload definition 5–13
workload set definition 5–10

D

data aggregation, performing 6–27
data collection
control 9–1
description 1–6, 1–8
of IOTRACE data 9–25
of MAPPER log data 9–29
of MIPS metering data 9–9
of Open Distributed Transaction Processing data 9–14
of SIP data 9–2
of System Activity Monitor data 9–5
of system log data 9–33
of TPM log data 9–17
processing SIP data 1–8
starting 9–1
storing data 1–8
tqctl runstream 9–1
data display
description 1–6
displaying reports 1–6
requesting data 1–6
retrieving data 1–6
data storage, **tqusrprb** probe 9–45
data value 2–3
database
aggregation set 1–10, 2–1
creating 10–3
data value 2–3
deleting 10–3
description 1–10, 2–1, 10–1
disk space requirements 10–4
files 10–2
listing 10–4
structure 2–1
table data 2–4
database administration
backup 10–8
clearing locks 10–7
creating
additional databases 10–3
aggregation set 6–10, 6–16
derived statistic 8–8
workload definition 5–13
workload set definition 5–10
deleting
aggregation set 6–15
databases 10–3
derived statistic 8–17
workload definition 5–18
workload set definition 5–12
displaying
aggregation set 6–14
derived statistic 8–16
workload definition 5–16
workload set definition 5–11
estimating disk space 10–4
exporting
derived statistic definition 8–15
workload definition 5–17
importing
derived statistic definition 8–15
workload definition 5–17
listing
aggregation set 6–20
databases 10–4
parameters 10–4
maintaining
aggregation sets 6–8
alarms 7–1
tables 11–3
thresholds 7–1
workloads 5–8
modifying
aggregation set 6–14
derived statistic 8–16
workload definition 5–16
workload set definition 5–11
performing data aggregation 6–27
recovery 10–8
removing aggregation set 6–19
saving derived statistic 8–13
dedicated Java processor statistics A–14
default threshold definitions 3–4
changing 3–5
using 3–4
defining
named pipe connection 4–6
retention specifications for tables 11–9
socket connection 4–5
socket session 4–1
SSL connection 4–6
definition rules for workloads 5–2
deleting
aggregation set 6–15, 6–19
an alias 8–13
collection period for tables 11–8

derived statistic 8–17
records from tables 11–2, 11–16
retention specifications for tables 11–14
threshold definitions 7–19
workload definition 5–18
workload set definition 5–12
deregistering tables 11–20
derived statistics
alias definition 8–2
creating 8–8
creating thresholds for 7–14
defining an alias 8–11
definition file format 8–18
deleting 8–17
description 2–6
displaying 8–16
exporting definitions 8–15
exporting definitions to a file 8–18
formula definition 8–2, 8–3
hierarchy 8–6
identification of 8–1
importing definitions 8–15
loading definitions 8–21
maintenance 8–5
modifying 8–16
parameter references 8–2
parameter substitution 8–2
saving 8–13
TeamQuest Performance Software Administration 8–4
tqzip administration 8–17
unloading definitions 8–18
updating thresholds for 7–16
using **tqrpt** for reporting D–5
viewing thresholds for 7–18
workload performance A–51
disk space requirements 10–4
displaying
aggregation set 6–14
derived statistic 8–16
tables 11–4
workload definition 5–16
workload set definition 5–11

E

encrypted connections
TQENC program I–10
user-id and password I–10
using I–1
encryption in TQTCP I–10
errors for TQTCP I–9
exporting
data using **tqrep** E–12
derived statistic definitions 8–15, 8–18
threshold definitions 7–21
workload definitions 5–17, 5–19
extracting statistical data F–1

G

general section, report definition file C–3

H

hierarchy, derived statistic 8–6
host software
data collection by 1–8
description 1–8
probe 1–8
host-based reporting
aggregation set data D–1
table data E–1

I

importing
derived statistic definition 8–15
workload definition 5–17
installing **tqusrprb** probe 9–47
IOTRACE data collection
description 9–25
parameters A–44
reports B–13
starting 9–25
I/O activity statistics A–24
I/O reports B–2

K

key names
description 2–2
viewing 8–14

L

listing
  aggregation set 6–20
  collection periods for tables 11–6
  databases 10–4
  parameters 10–4
  retention specifications for tables 11–11
loading
  derived statistic definitions 8–21
  threshold definitions 7–24
  workload definitions 5–22

M

maintaining
  alarms 7–1
  tables 11–1, 11–3
  thresholds 7–1
management console configurations
  Computer Associates configuration J–9
  HP OpenView configuration J–7
  overview J–6
  Tivoli configuration J–7
MAPPER log data collection
  description 9–29
  file input to **tqmpr** 9–31
  parameters A–45
  reports B–14
  specifying workload definitions 5–1
  starting 9–29
memory
  reports B–7
  statistics A–18
MIPS metering data collection
  description 9–9
  starting 9–9
modifying
  aggregation set 6–14
  derived statistic 8–16
  primary retention specification for tables 11–12
  workload definition 5–16
  workload set definition 5–11

N

named pipe connection, defining 4–6

O

Open Distributed Transaction Processing
  Client table information A–64
  description of data collection 9–14
  Service table information A–64
  starting data collection 9–14
  System table information A–61

P

paging statistics A–22
parameter
  data value 2–3
  description 2–2
  hierarchy of structure 2–2
  key names 2–2
  listing 10–4
parameters
  Capacity On Demand systems A–43
  IOTRACE A–44
  MAPPER log A–45
  MAPPER log probe A–53
  Open Distributed Transaction Processing A–61
  SIP A–2
  SIP probe A–51
  system activity monitor A–58
  system log A–46
  system log probe A–54
  TPM log A–41
parameter-group section C–29
password encryption I–10
performance database, backup and recovery 10–8
performance monitor statistics A–9
performing
  data aggregation 6–27
  derived statistic maintenance 8–5
probe
  description 1–8
  tqbsp 1–8, 9–2
  tqiot 1–9, 9–25
  tqmips 9–9
  tqmpr 1–9, 9–29
  tqodtp 1–10, 9–14
  tqrtpm 1–9, 9–17
  tqsam 1–9, 9–5
  tqsys 1–10, 9–33
  tqtblprb 1–10, 9–38

tqtpm 1–9, 9–17
tqusrprb 1–10, 9–38
processor call, for TQTCP I–4
processor reports B–6, B–8
product capabilities 1–5
program load statistics A–18

Q

qualifying parameters 8–12
queue statistics A–39

R

recovery of performance database 10–8
registering tables 11–2, 11–18
removing aggregation set 6–19
reorganizing table files 11–26
report
definition directory B–1
description of predefined reports B–1
history 1–11
host-based D–1
monitor 1–11
user-defined 1–11
report definition file
description C–2
example C–1
for **tqrep** program E–3
format C–2
general section C–3
notation conventions for C–2
parameter-group section C–29
resource 2–2
retention periods for tables 11–2
Retention Specification Menu of **tqzip** 11–9
retention specifications
adding 11–13
deleting 11–14
description 11–9
listing 11–11
modifying the primary retention 11–12
setting start time 11–15
Runs Table, System Activity Monitor A–58

S

saving derived statistic 8–13
security considerations, for TQTCP I–10
sending SNMP traps J–1
Server table information A–62
Server Table, Open Distributed Transacton Processing A–62
Service Table, Open Distributed Transaction Processing A–64
session timeout messages 4–7
sessions
named pipe connection 4–6
socket connection 4–5
SSL connection 4–6
sessions, socket connection 4–1
setting retention start time 11–15
Single Point Operations (SPO)
considerations for using with TeamQuest Baseline 7–32
SIP data collection
description 9–2
file input to **tqbsp** 9–4
starting 9–2
terminating 9–3
SIP parameters
category groups A–2
dedicated Java processor statistics A–14
I/O activity A–24
memory statistics A–18
paging statistics A–22
performance monitor statistics A–9
processor statistics A–7
program load statistics A–18
queue statistics A–39
system statistics A–5
XPC statistics A–36
SIP reports
I/O reports B–2
memory reports B–7
processor reports B–6, B–8
summary reports B–11
system reports B–10
SNMP traps
alarm administration J–2
CMS 1100 configuration J–2
Computer Associates configuration J–9
content J–4
HP OpenView configuration J–7
management console configurations J–6
mappings J–4

overview J–1
sending J–1
special considerations J–12
syntax J–2
TeamQuest MIB definition J–9
Tivoli configuration J–7
variable fields J–5
socket connection, defining 4–5
sockets
connections I–11
defining
connections 4–1
interface overview I–1
special considerations for SNMP traps J–12
specifying workload definitions 5–1
SSL connection, defining 4–6
storing table data 9–57
summary reports B–11
syntax of SNMP traps J–2
system
reports B–10
statistics A–5
System Activity Monitor
Runs table information A–58
Transactions table information A–59
System Activity Monitor data collection
description 9–5
parameters A–58
starting 9–5
terminating 9–6
system log data collection
description 9–33
file input to **tqsys** 9–35
parameters A–46
reports B–15
specifying workload definitions 5–1
starting 9–33
System Table, Open Distributed Transaction Processing A–61

T

table data
description 2–4
host-based reporting E–1
storing 9–57
tqsam probe 9–8
tables
adding
collection period 11–7
retention specifications 11–13
administering collection periods 11–4
applying
collection period changes 11–9
retention specification changes 11–15
Client A–64
collection periods 11–1
deleting
collection periods 11–8
records 11–2, 11–16
retention specifications 11–14
deregistering 11–20
displaying 11–4
listing
collection periods 11–6
retention specifications 11–11
main menu 11–3
maintaining 11–1, 11–3
modifying primary retention specification 11–12
registering 11–2, 11–18
reorganizing table files 11–26
retention periods 11–2
Runs A–58
Server A–62
Service A–64
setting retention start time 11–15
specifications
file format 11–21
System A–61
tqzip administration 11–3
Transactions A–59
turning on and off data collection 11–9
Tables Menu of **tqzip** 11–3
TeamQuest Baseline
aggregation set description 3–3
alarm handling 3–3
capabilities 1–5
considerations for using with SPO 7–32
data collection 1–6
data collection control 9–1
data display 1–6
database 1–10, 2–1
default threshold definitions
changing 3–5
description 3–4
using 3–4
description 1–4
host software description 1–8

host-based reporting D–1
overview of processing 1–5
probe 1–8
tqdata 1–11
tq-ini element 3–1
TeamQuest MIB definition, for SNMP traps J–9
TeamQuest Online
data collection 1–6
data display 1–6
host software description 1–8
overview of processing 1–5
probe 1–8
tqdata 1–11
tq-ini element 3–1
TeamQuest Performance Software Administration
additional sessions 4–6
aggregation set tasks 6–8
derived statistic tasks 8–4
establishing a session 4–1
introduction 4–1
socket connection 4–1
terminating 4–8
warning screens 4–7
workload tasks 5–8
TeamQuest View
alarm messages displayed by 7–30
data display 1–6
description 1–11
predefined report descriptions B–1
retrieving data 1–6
terminating
administration session 4–8
SIP data collection 9–3
System Activity Monitor data collection 9–6
tqbsp 9–3
tqsam 9–6
threshold definitions
changing defaults 3–5
defaults 3–4
using defaults 3–4
thresholds
checking 7–21
creating 7–6
creating for derived statistics 7–14
deleting 7–19
exporting definitions to a file 7–21
loading definitions 7–24
maintaining 7–1
reusing threshold definitions 7–11
turning on and off threshold checking 7–25
unloading definitions 7–21
updating definitions 7–12
updating for derived statistics 7–16
viewing 7–13
TPM log data collection
description 9–17
file input to **tqtpm** 9–21
parameters A–41
reports B–17
starting 9–17
tqagg program
command line format 6–27
description 6–27
tqalm program
alarm messages 7–30
alarms handling 7–28
command line format 7–28
description 7–28
processor call statement 7–28
user-defined alarm handling 7–31
viewing alarm messages 7–27
tqasa program
command line format 6–16, 6–19, 6–20, 6–26
converting expanded aggregation sets 6–26
creating an aggregation set 6–16
listing aggregation sets 6–20
removing an aggregation set 6–19
tqbsp program
abnormal termination 9–3
description 1–8, 9–2
parameters collected by A–2, A–51
processor call statement 9–2
SIP file input 9–4
tqctl runstream 9–1
tqdata description 1–11
tqdbu utility
creating databases 10–3
listing databases 10–4
TQENC program for encryption I–10
tqgetm program F–1
tqiot program
description 1–9, 9–25
parameters collected by A–44
processor call statement 9–26
workload characteristics 9–28
tqmips program
abnormal termination 9–11
description 9–9

parameters collected by A–43
processor call statement 9–9
terminating data collection 9–11
trail selection 9–11
tqmpr program
changing default log file names 9–32
description 1–9, 9–29
log file input 9–31
MAPPER log file input 9–31
parameters collected by A–45, A–53
processor call statement 9–29
workload characteristics 9–33
tqodtp program
abnormal termination 9–15
description 1–10, 9–14
parameters collected by A–61
processor call statement 9–14
table data 9–16
terminating data collection 9–15
tqrep program
command format E–2
default reports E–12
description E–1
exporting data E–12
report definition file E–3
tqrorg utility
command line format 11–28
description 11–26
reorganizing table files 11–26
tqrpt program
condensed output D–4
description D–1
reporting derived statistics D–5
tqrtpm program
description 1–9
processor call statement 9–18
terminating data collection 9–19
tqsam program
abnormal termination 9–7
description 1–9, 9–5
parameters collected by A–58
processor call statement 9–5
table data 9–8
workload characteristics 9–7
tqsys program
description 1–10, 9–33
parameters collected by A–46, A–54
processor call statement 9–34
system log file input 9–35
workload characteristics 9–37
tqtblprb probe
collecting and storing data 9–62
command line format 9–62
creating a table 9–62
data fields 9–49
description 1–10, 9–48
example for collecting and storing data 9–58
field definition 9–49
field definition guidelines 9–49
group key fields 9–49
input format 9–48
primary key fields 9–49
removing a table 9–62
table data value formats 9–65
table definition 9–49
table specification 9–49
writing a probe for table data 9–48
TQTCP executable
background run I–3
configuration element I–4
description I–2
errors I–9
processor call I–4
security considerations I–10
using encryption I–10
tqtpm program
description 1–9, 9–17
log file input 9–21
parameters collected by A–41
processor call statement 9–20
workload characteristics 9–23
tqunlk program
description 10–7
processor call statement 10–7
tqusrprb probe
data storage 9–45
description 1–10, 9–39
installing 9–47
PRBDATA file input format 9–39
processor call format 9–39
TQARG file input format 9–39
writing a probe for performance data 9–39
tqzip program
adding
collection period 11–7
retention specifications 11–13
Alarms and Thresholds Menu 7–5
applying
collection period changes 11–9
retention specification changes 11–15

Collection Period Menu 11–4
creating
derived statistic threshold definitions 7–14
threshold definitions 7–6
deleting
collection periods 11–8
records 11–16
retention specifications 11–14
threshold definitions 7–19
deregistering a table 11–20
displaying a list of tables 11–4
exiting
Collection Period Menu 11–9
Retention Specification Menu 11–15
listing
collection periods 11–6
parameters 10–4
retention specifications 11–11
loading
derived statistic definitions 8–21
threshold definitions 7–24
workload definitions 5–22
maintaining
alarms 7–1
thresholds 7–1
modifying primary retention specification 11–12
notifying probes of threshold changes 7–21
processor call format H–1
registering tables 11–18
Retention Specification Menu 11–9
reusing threshold definitions 7–11
setting retention start time 11–15
threshold checking 7–21
turning on and off
data collection 11–9
threshold checking 7–25
unloading
derived statistic definitions 8–18
threshold definitions 7–21
workload definitions 5–19
updating
derived statistic threshold definitions 7–16
threshold definitions 7–12
viewing
alarm messages 7–27
derived statistic thresholds 7–18
thresholds 7–13
tq-ini element 3–1
Transactions Table, System Activity Monitor A–59
turning on and off data collection 11–9

U

unloading
derived statistic definitions 8–18
threshold definitions 7–21
workload definitions 5–19
updating
derived statistic threshold definitions 7–16
threshold definitions 7–12
user data probes
tqtblprb for collecting table data 9–38
tqusrprb for collecting performance data 9–38
writing a user probe for performance data 9–39
writing a user probe for table data 9–48
user interface, description 1–11
user-id encryption I–10
using encryption I–1

V

viewing
alarm messages 7–27
derived statistic thresholds 7–18
long key names 8–14
thresholds 7–13

W

warning screen 4–7
workload definition language 5–2
workload maintenance
creating
workload definition 5–13
workload set definition 5–10
deleting
workload definition 5–18
workload set definition 5–12
displaying
workload definition 5–16
workload set definition 5–11
exporting workload definition 5–17

importing workload definition 5–17
modifying
workload definition 5–16
workload set definition 5–11
workload performance derived statistics A–51
workload performance reports B–19
workloads
definition
file format 5–19
language 5–2
rules 5–2
description of concepts 2–4
exporting definitions to a file 5–19
IOTRACE probe 9–28
loading workload definitions 5–22
MAPPER log probe 9–33
specifying workload definitions 5–1
system log probe 9–37
TeamQuest Performance Software Administration 5–8
TPM log probe 9–23
tqzip administration 5–18
unloading workload definitions 5–19
workload definition language 5–1
workloads, characteristics
for IOTRACE probe 9–28
for MAPPER log probe 9–33
for System Activity Monitor probe 9–7
for system log probe 9–37
for TPM log probe 9–23
workstation software, description 1–11

X

XPC statistics A–36